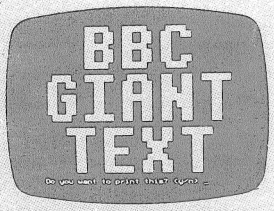
£1.00

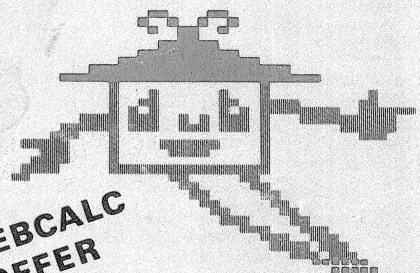
BEEBUG

FOR THE BBC MICRO

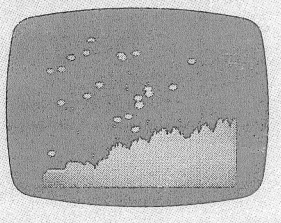
Vol 2 No 4 AUG/SEPT 83



BIG CHARACTERS

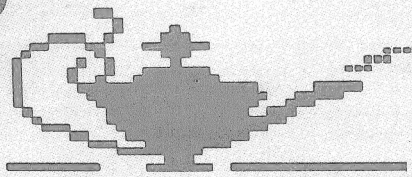


BEEBCALC OFFER

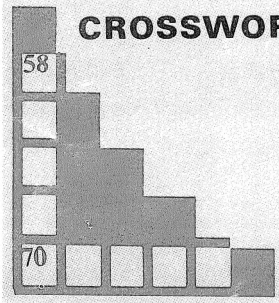


MARS LANDER

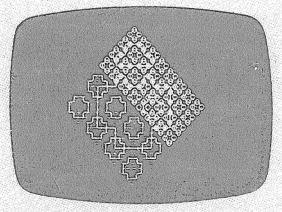
MICRONET 800 REVIEWED



*SPACE LORDS
*UNDERSTANDING ASCII
*MICROTAX REVIEWED
*BUILD YOURSELF A LIGHT PEN
*CONTACT POINTS FOR THE BEEB



CROSSWORD



SPIDERS WEB

BRITAIN'S LARGEST COMPUTER USER GROUP
MEMBERSHIP EXCEEDS 19,000

EDITORIAL

We would like to thank those members who introduced themselves at our stand at the London "Computer Fair" at Earls Court in June, it was nice to meet you. The next major show is the "Acorn User" show in late August, this is almost certainly where the Acorn Electron will be launched. The Model-A is being discontinued, this ties in with the launch of the Electron.

NEW SUBSCRIPTIONS & BACK ISSUES DEPARTMENT

As part of a streamlining of our operation, we have moved the Subscriptions and Back Issues Department to the extremely efficient mailing agency currently used for software. Please make sure that you use the correct address for our various offers, and it helps if you do NOT combine orders for different addresses, as this delays matters considerably.

HARDWARE PROJECT

On page 19 we have an article entitled "Build Yourself a Light Pen". This is great fun, and good for experimenting with. This can be recommended as an ideal first hardware project, so do have a go.

BEEBUG OFFERS

We have looked at the spread-sheet program "Beebcalc" produced by Computer Concepts, and in our estimation it comes up to the standards of Wordwise. We have managed to negotiate a similar discount for members. We have also secured a further discount on Wordwise, and pass this directly on to members. For full details of both offers see page 42. We will carry a detailed review of Beebcalc in our next issue.

Following the launch of the BEEBUG "Magazine Cassette", we are pleased to announce an annual subscription to this. See the back cover for details.

EXMON is now available in EPROM, with the great advantages that there is no need to load it, and it takes up none of the Beeb's precious user memory. See the supplement for details of this offer.

THE "AUSTRIAN CONNECTION"?

Thanks to those of you who responded to our plea and sent in User Guide errors. We have passed these on to Acorn who are incorporating them in a German version of the User Guide, this is to go with the launch of the BBC Micro in Germany.

1.2 OS and BASIC II

The 1.2 OS has proved remarkably bug free, and it is unlikely that Acorn will release a new OS for some considerable time (if ever). It therefore seems sensible to recommend (strongly) that all users upgrade their machines to 1.2 at the earliest opportunity. We make this recommendation essentially because the 1.2 has a number of enhancements that are being made use of in recent software, and hardware add-ons, and this means that the amount of software that will run on a 0.1 system will become increasingly limited. We have sold many thousands of these ROMs, and very few people have had any problems in fitting it themselves.

A number of members have asked if we will be offering a similar deal on the Basic II chip. The answer is that Acorn have not yet decided how they will be marketing this chip, though they are considering marketing it as a separate entity.

OTHER NEWS

Acorn tell us that they now have a booklet entitled "Sideways ROMs". This is available from Cash Sales on 0223 245200 and can be ordered by credit card.

M-TEC Computer Services now have a full "Torch BBC Basic" (v1.91) available at £110 plus VAT. If you had an earlier version, then the cost is only £15 when your old disc is returned. Contact: M-TEC Computer Services on 0603 870620.

Please remember this is the Aug/Sept issue - no magazine next month. **Sheridan Williams**

HINT WINNERS

This month's hint winners are Matthew Rapier who wins £10, and Bjorn Grand who wins £5. Keep those hints rolling in please.

BEEBUG MAGAZINE

GENERAL CONTENTS

PROGRAMS

HINTS, TIPS & INFO

ADDITIONAL ITEMS ON MAGAZINE CASSETTE

Epson 8-Tone Screen Dump - Both an article and the program are on the magazine cassette.

"Assembly Language Programming for the BBC Microcomputer" by Birnbaum. This book contains no index, we supply a very comprehensive one here. We reviewed this excellent book in Vol.1 No.9 P.17

Crossword Answers - Advance publication of the answers to our crossword. Answers will be printed in the magazine in the next issue.

UNDERSTANDING ASCII

by Sheridan Williams

Sheridan explains the essentials of ASCII codes.

Like all computers, the Beeb uses a number code to represent each of the printable characters. This is based on the so-called ASCII code (American Standard Code for Information Interchange). For example, the ASCII code for the letter "P" is 80. You can print the character P in two ways: either by telling Basic to: PRINT"P" or to: PRINT CHR$80. The CHR$ simply informs Basic to use the character code, and print the corresponding character.

You can find out the ASCII code of any given keyboard character by using the ASC function. Thus:

```
PRINT ASC"P"
```

will print the number 80.

The following program will print the ASCII code of any character input from the keyboard:

```
10 INPUT"Press a key, then 'RETURN";A$
20 PRINT"Its ASCII code is ";ASCA$
30 GOTO 10
```

If you experiment with this you will see that no codes are printed outside the range 32-127.

ASCII CODES IN THE RANGE 32-127

On the BBC Micro 256 codes are used. Codes 32-127 are used to represent the following:-

Codes	Characters	Codes	Characters
32- 47	sp - /	91- 96	[- `
48- 57	0 - 9	97-122	a - z
58- 64	: - @	123-126	{ -
65- 90	A - Z	127	delete

'sp' represents the 'space' character. The three shaded ranges - the numbers 0 to 9, and the upper and lower case letters form three important blocks.

In practice, a few codes produce different characters depending on whether you are using mode 7 or one of the other modes. Complete tables of ASCII codes can be found in the User Guide on pages 486-492. The following program will display the codes from 32 to 127 and their characters.

```
10 VDU 14: REM Turn paging on
20 FOR code%=32 TO 127
30  PRINT ;code%;TAB(5);CHR$code%
40 NEXT code%
50 VDU 15: REM Turn paging off
```

Lines 10 and 50 simply turn on and off the paging feature, so that the program pauses after each screenful. Try running the program first in mode 7, then in mode 4, to see the differences.

The 'Monkeys typing Shakespeare' gives a simple example of the use of ASCII codes in conjunction with random numbers.

ASCII CODES BELOW 32

These are known as "Control codes". They generate invisible characters which have certain special effects. For example, try PRINT CHR$12 and you should see that the screen is cleared. PRINT CHR$7 will produce 'beep'. Other characters are used as printer control characters. CHR$2 for example turns a printer on. The effect of using such codes is also tabulated in the User Guide. It is more common to issue these as VDU commands, so VDU 7 is exactly the same as PRINT CHR$7; One advantage of using a VDU call is the ability to string several characters together. Thus VDU 13,13,7 will execute two carriage returns followed by a 'beep'.

Unlike the first program in this article, the program below will give you the ASCII code for a larger range of keys selected from the keyboard, including those obtained by pressing the CTRL key in conjunction with another key.

```
10 REPEAT
20  ch$=GET$: ch=ASCch$
30  IF ch>31 AND ch<128 PRINT ch$;
40  PRINT TAB(2);ch
50 UNTIL FALSE
```

This program will print the ASCII codes for the standard keyboard (Not 'Escape', 'Break', function or cursor keys). If the ASCII code lies between 32 and 127, then the program will print its character equivalent too.

ASCII codes are well thought out, and the layout of many of the keys is determined by them. For instance, using the program just given, see if you can find a relationship between CTRL-A, Upper case-A and Lower case-a. You will see the relationship more clearly if you print the ASCII codes in hexadecimal rather than in decimal. You can do this by replacing line 40 in the previous program by:-

```
40 PRINT TAB(2);~ch
```

Note the ~ character before the ch, it instructs the computer to print the numbers in hexadecimal (base 16), rather than decimal (base 10). For a brief explanation of hexadecimal code see BEEBUG Vol.1 No.8 P.16.

ASCII CODES ABOVE 127

ASCII codes above 127 are reserved for Teletext characters in mode 7, and are available for you to define in other modes, these have been mentioned in previous issues of BEEBUG in particular 'TELETEXT' Vol.1 No.4 P.7 and 'GRAPHICS' Parts 1 and 2 in Vol.1 No.2 P.9 and Vol.1 No.3 P.11. For example, go into mode 7, (eg. by executing Break) then type VDU 132,157,129:PRINT "RED TEXT ON BLUE BACKGROUND"

As further illustration the following program allows you to see the effect of keyboard characters in the Teletext graphics mode. To enter the Teletext graphics mode you must use an ASCII code in the range 145 to 151 - depending on the colour (see BEEBUG reference card).

```
10 MODE 7
20 INPUT A
30 PRINT CHR$129;CHR$A;CHR$145;CHR$A
40 GOTO 20
```

The above program allows you to type in a number when the question mark appears, and it will the show you both the ALPHA character and the GRAPHIC character for that given code. For best effect in GRAPHIC characters you should use the ASCII codes in the range 95-126 and 160-191 as shown on pages 488/489 of the User Guide.

GRAPHICS CHARACTERS

To create your own graphics characters you need to be in any mode apart from mode 7. Then you tell the computer which ASCII character you wish to define. At its simplest, you can define any ASCII character from 224-255. This is done using the VDU23 statement (See the reference above for more details on precisely how to define the characters).

The following program sets mode 6, and then defines two characters 224 and 255 as two space invaders (lines 20-30). It then displays them repeatedly:

```
10MODE6
20VDU23,224,60,126,219,255,126,60,36,66
30VDU23,255,129,90,60,90,126,60,66,36
40PRINT CHR$224;CHR$255;
50GOTO 40
```

(NOTE: Although modes 3 and 6 are not graphics modes, you can still print user-defined graphics in them).

MONKEYS TYPING SHAKESPEARE

There is a well known statistical saying, that if you sit enough monkeys at enough typewriters, and leave them for long enough, they will eventually type the complete works of Shakespeare! Here we will make the computer perform the same task. It involves the generation of random characters, rather than random numbers; this can be achieved by using the ASCII codes for the letters of the alphabet. Essentially we are using a knowledge of ASCII codes to allow us to produce letters of the alphabet from a series of (random) numbers.

Here is the method that we are going to adopt (the algorithm):

1. Generate a random number between 97 and 122 inclusive. (ASCII codes for the letters a-z)
2. Print the character equivalent to this number (lower case only).
3. Occasionally print a space.
4. Return to step 1.

The program to do this is:

```
10 REPEAT
20  rand letter%=RND(26)+96
30  PRINT CHR$rand letter%;
40  IF RND(1)<0.3 PRINT CHR$32
50 UNTIL FALSE
```

The computer is not really like the monkeys, because the program produces only letters, whereas monkeys could press any key. To limit monkeys to just letters either assumes that the monkeys have some intelligence, or they are using a special typewriter. You can have endless fun playing with this program trying to make it generate more sensible text.

THE WATFORD ELECTRONICS DISC FILING SYSTEM REVIEWED

by Colin Opie

Several factors such as the decreasing cost of disc systems, the shortage of official disc upgrades, and Acorn's policy of only supplying the DFS to dealers has prompted a number of suppliers to try and provide alternatives disc filing systems. One such supplier is Watford Electronics referred to from now on as WE. We have had a pre-release version of their DFS together with a manual here at BEEBUG for some time, and report below on our findings.

PACKAGING

The DFS will be supplied in a 16k EPROM (The Acorn DFS is 8k) and will have a manual to accompany it costing £7.50. The ROM fits as usual into one of the sideways sockets, and the manual is there to tell you all you need to know about using the Disc System. Cost will be around £42 for the DFS, and around £85 for the complete upgrade kit. Watford Electronics will exchange their DFS for the Acorn DFS for £38. All prices are exclusive of VAT.

FEATURES

The manual is quick to point out that the WE DFS is an enhanced version of the Acorn system. Indeed all the normal DFS commands are available (see BEEBUG Vol.1, No.10, P.25 for a list and explanation of these). What is impressive about the WE DFS is the extras which have been included. It is not possible, or necessary, to document these in full but the following is a summary of the features:

1) An extra utility '*EDIT' is included which is a disc sector editor program. (Uncannily similar in looks to the one in BEEBUG Vol.2 No.3!).
2) A disc formatting program is part of the DFS (ie. not a file on a utility disc), as is a disc verifier. They can be accessed via *FORMnn and *VERIFY commands. An interesting point about the first of these two is that it gives you the option of *FORM80, *FORM40 and *FORM35, (for 80, 40, and 35 track discs , as does the BEEBUG Formatter - BEEBUG Vol.1 No.10). You can also ask for a '62 file directory' as opposed to the normal 31 files allowed on the Acorn DFS. (More on this later in the article).
3) Four extra DFS commands exist, *MLOAD, *MRUN, *MOVE and *WORK. *MLOAD and *MRUN are really utilities for automatically relocating programs which would normally be too big to run on a disc machine (eg. a program which was originally a cassette program). *MOVE is much the same as *COPY (standard DFS) but gives an extra level of choice over which files are copied by requesting a yes/no response on all files which match the file specification. In this sense it is similar to *WIPE except that *WIPE is concerned with deleting files, while *MOVE is concerned with copying them. *WORK is very useful in that in enables you to specify a 'work' filename. After using this command you can simply type eg. SAVE"" , and the DFS will insert the name for you from the *WORK specification. Explicit SAVE and LOAD commands are not affected by *WORK. Also the catalogue listing displays the current *WORK specification.

EXTRA HELP MESSAGES

Two additional '*HELP' messages are supplied and accessed through *HELP FILES, and *HELP SPACE. The first of these will list any 'open' files - useful for debugging Basic file handling programs. The second will list all the 'gaps' on a specified disc so that you can see if it is worth compacting it (*COMPACT) in order to gain more storage space.

READING 40 TRACK DISCS

Many of the drives available on the market are either 40 track or 80 track, it costs you extra if you want to have a 'switchable' unit. The WE DFS goes as

far as it can in this respect by supplying a *FX110 command which enables you to read a 40 track disc in an 80 track drive.

62-FILE CATALOGUES

When formatting a disc the WE DFS will allow you to create a catalogue which can hold 62 file names, rather than the normal 31. When using 80 track drives this is extremely useful as very often you run out of catalogue space before you run out of storage space. Discs formatted in this way clearly cannot be used properly by other systems but the facility still exists to create 'standard' formatted discs so no practical problems tend to arise.

CONCLUSION

Watford Electronics have obviously put a lot of work into this DFS and there are certainly some welcome additions to the capabilities of the standard DFS. The cost is comparable to that of a standard disc upgrade and this DFS therefore appears to be good value for money. The one problem with software of this type and scale however is its degree of robustness and lack of bugs. I have not found any real clangers yet and I am assured that all known bugs have been removed. The DFS has worked well for me during its use (though I have not tested all the machine code links and subroutines).

SORT COMPETITION RESULTS

WINNER OF THE SORT COMPETITION

Some time ago Sheridan Williams set a competition for the fastest and most useful machine code sort program submitted. We received only 6 entries and these were from:

M.J.Carter, R.H.Tracy, D.Fletcher, D.T.Blyth, C.Marshall, J.B.Simpson

The winner of the £50 prize is J.B. Simpson.

His was one of the three entries that would sort all three types of array variable, and was the only one that worked in all the tests applied to it. We hope to publish this program in the next issue together will a full description of how to use it.

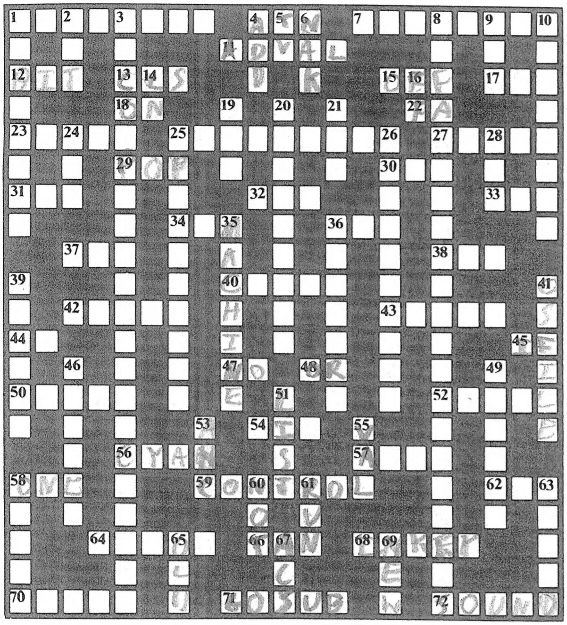
HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

COLOURED FILENAMES - Peter Thorne

Acorn's DFS permits the use of coloured filenames or disc titles, generated by the use of the Shift-Function keys. However the former will totally confuse the DFS when *CAT is produced, as it detects that the top bit of the first character of the filename is set and decides it has already output that name. This results in coloured filenames not being output by *CAT. It is thought that later versions of Acorn DFS will not allow the use of control characters in filenames, or even in the disc title. Watford Electronics DFS (see review in this issue) only permits their use in the disc title.

FORCING MEMORY CLEAR ON BREAK - T.G.E. Bromilow

*FX200,2 will cause BREAK to clear the entire contents of memory, from &400 to &7FFF, omitting the zeroth byte of each page. This could be useful for software protection.
(For more details on FX200 see BEEBUG Vol.2 No.3 P.34)



BEEBUG CROSSWORD

by Douglas Nunn

We present a crossword with a BBC Micro theme. The answers will be published in the next issue. If you would like to make up a crossword for publication please keep the total number of clues below 40; mark your envelope "CROSSWORD" and post it to the editorial address. We will pay £25 for any crossword published.

ACROSS CLUES

1. Confused crumpet has nothing else (8)
4. Inverse of 66 across (3)
7. What a computer's made of (8)
11. 4 down & 55 down (well, almost!) (5)
12. Position On Screen? (3)
13. Control-L (3)
15. Opposite of 18 across (3)
17. Integer division (3)
18. Basic equivalent of CASE X OF ----,----,---- (2)
22. 250 in hex (2)
23. Bottom of variable storage (5)
25. Quarrels over parameters (9)
27. REPEAT...----- (5)
29. eg.Jam,Who,etc....(3)
30. Royal line in error (3)
31. Machine code comparison (3)
32. 3 down (abbreviation) (3)
33. Pull processor register from stack (3)
34. Ignore this line! (3)
36. Non-maskable interrupt (abbreviation) (3)
37. Length of a disc file (3)
38. (pass*2-list) (3)
40. Similar, often identical to 12 across (5)
42. See 25 down (5)
43. Output (5)
44. Unlucky character (2)
45. Basic command for a conditional branch (2)
47. Backwards 18 across (2)
48. Either...(2)
50. ASCII 33 (5)
54. Man who lives at &FD00 - &FDFF (3)
56. The flasher's gone red? No, quite the opposite! (4)
57. Command mode statement that can be silly! (4)
58. ASCII 49 (3)
59. Next to 1\down\-\what every user should have! (7)
62. Inverse of 14 down (3)
64. Mistake? It sounds Irish! (5)
66. Hide from trigonometry? (3)
68. Enter from keyboard... (5)
70. My pen leaked on the keyboard (5)
71. Command virtually made obsolete by PROC (5)
72. Ouput to speaker (5)

DOWN CLUES

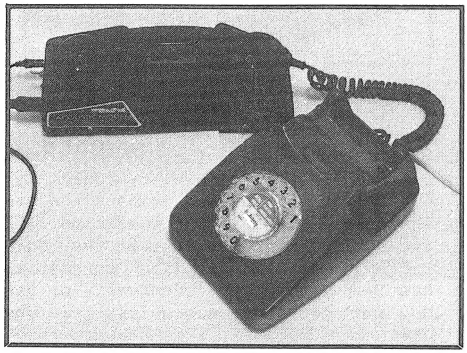
1. Security for headgear (4,4)
2. Stirling worth £5.85 (3)
3. Ferranti took their time with it! (11,5,5)
4. Plus (3)
5. Where you can stick your UHF output (2)
6. VDU21 (ASCII abbrev p378 U.G.) (3)
8. VDU20 (7,7,7)
9. CHR$38, CHR$43 - both mean. ..(3)
10. Container for ASCII 65 to 90 (8)
14. Real wood? (2)
16. Very loud 255 (2)
19. Function returning -1,0,1 (3)
20. Last six parameters of 10 down (9)
21. Unknown quantity (3)
24. Way of navigating through memory (3)
25. and 42 across: Feature of the 6502, something it has many of (10 and 5)
26. Cornflakes and sweet red wine (6,4)
28. Usually hangs around with 23 across (3)
35. ------- code (7)
36. Fishing job (7)
39. Key to use in prison (6)
41. Routine at &FFDD (6)
46. Characters used to print in hex (6)
49. System for linking Acorns (6)
51. To display current program (4)
53. To add one to (3)
55. Opposite of STR$ (3)
58. Routine at &FFF7 to interpret command line (5)
60. Tie up bitwise flip-flop (3)
61. Execute Basic program (3)
63. ASCII 96 on Beeb (5)
65. Often defined on key 10 (3)
67. Trigonometry used to kill dwarves? (3)
69. Opposite of 65 down (3)

MICRONET 800 REVIEWED

by David Graham

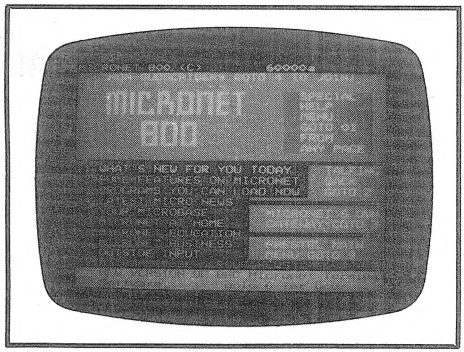
A Prestel/Micronet terminal for around £50 down and £1 a week. Is it too good to be true? David Graham Investigates.

Micronet 800 costs £56.00 to join. There is in addition a rental charge of £1.00 per week. You may also incur further charges if you access Prestel during office hours (or saturday before 1pm), or if you make use of charged pages, or the teleshopping facility. All charges apart from the joining fee are added to your quarterly telephone bill.



For your money you get a modem which takes a standard telephone handset, and which plugs into the RS423 port on the Beeb (Model B only). It is accompanied by software on cassette (at 300 baud only, unfortunately). To access Micronet, you first load the software, then dial one of the Telecom Prestel computers on your telephone, then place the handset on the modem. Once you have entered your personal identity number (supplied by Prestel) and your password, you are logged into Prestel and Micronet; giving you a potential 200,000 pages of information.

Each page is a full colour Teletext screen, and takes only a few seconds to download. As with Prestel, you can select any desired page by keying in the page number (you are provided with a printed index both for Prestel and for Micronet). Alternatively you can use the extensive menu facilities to find your way around. On Prestel you can always begin on page 1, and branch successively through subsequent indexes until you arrive at the desired topic.



Micronet is a subset of Prestel accessible only to Micronet members. It offers an interesting range of facilities for the BBC user. There are regularly updated pages, a classified ads section, a letter answering service, a new hints and tips section and a telesoftware section. This latter facility, together with Prestel's Mailbox service and a banking service called Homelink, exploits the medium admirably, and each deserves a closer look.

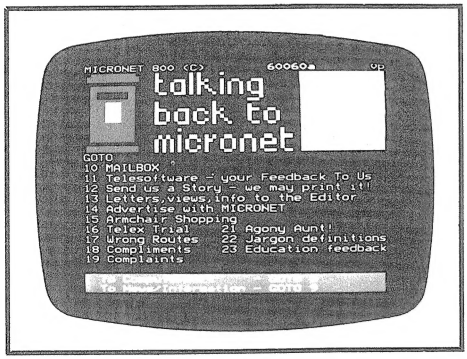
MAILBOX

The Mailbox area of Prestel allows you to send messages of all kinds to any Prestel user, including comments to the Micronet team itself. You do this by selecting a response frame, and typing in and editing text. If you choose to send the frame, the addressee (you get his Mailbox number from an on-screen directory) will be informed that a letter is waiting for him the next time he logs on or off. He can then access the frame which you created. The full Mailbox service is presently only available on the Prestel's Enterprise Computer (London Area), but other computers are expected to offer this soon. By a similar process it is possible to do teleshopping (with a credit card

number) or to order free brochures etc. These latter facilities are already available generally.

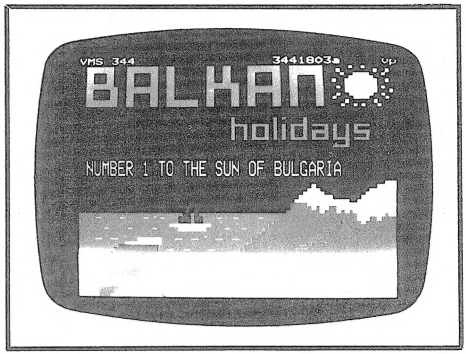
TELESOFTWARE

The system software supplied with the Micronet 800 kit allows you to save selected Teletext screens to tape or disc, or to send them to a printer. For this latter facility you must have a parallel connected printer (the serial port is used for the modem). Even more interesting is the facility to download software directly, at the press of a key. Some software on Micronet is free, and the quality of this is varied, others (such as a range of Acornsoft products) is charged to your next telephone bill. In either case, when you choose to download, the screen fills with mostly meaningless characters for a number of frames (a frame for every 1k of program approximately). Error checks are made on each frame, and if there is an error, the frame is automatically re-sent. Once downloaded, you can press a key to save the program to cassette or disc, from where it can be chained in as normal at a later date.



HOMELINK

This is a new facility currently offered jointly by Prestel, the Nottingham Building Society, and the Bank of Scotland. It allows Prestel and Micronet users to access deposit and current accounts, pay bills and call up on-screen statements, all from their keyboard at home. To join the scheme you must have at least £1000 invested with the Nottingham Building Society (but full interest is given on this). We shall be investigating this service, and will report back. In the meantime further details can be obtained on page 444 of Prestel, or by dialling the operator, and asking for Freefone HOMELINK. Note that investors with more than £3000 on deposit, receive a modem from Homelink free of charge.



Courtesy of VMS. Frame 344

EVALUATION

My privately ordered Micronet apparatus took around 5 weeks to arrive, which was a little longer than I had hoped, but by no means unreasonable. I should perhaps add that the reason that we have taken so long to review Micronet is that the review kit that we were promised months ago has yet to arrive. Eventually we gave up, and ordered one privately. This unit worked first time around, though the Prestel first page which requests your identity number invariably fails to appear, and small dashes appear at the top of the screen. These are responses to the keyboard. If you press the keyboard a few more times the logging-on frame usually arrives.

Generally the system software works well, though it could be improved somewhat by making it automatically configure for various operating systems etc (the user must do this manually at present). The software could also make better use of the function keys on the new operating system so as to allow the user to program his own keys with his favourite page numbers, and with his identity number to automate the logging-on process.

The only real problem which I encountered was caused by noise on the telephone line. This can result in corrupted pictures, and can give problems with telesoftware; though the

error checking mechanism ensured faultless downloads in our tests. I personally found that a proportion of calls initially gave badly corrupted frames; though adjusting the position of the handset as suggested in the manual often solved the problem.

OVERALL

In my own estimation, and in that of other members who have contacted us, using Micronet is a useful and entertaining adjunct to the Beeb. It provides a wealth of facilities and a myriad of pages through which to browse. The great thing about it is that it is fully interactive, and that pages are regularly updated. Micronet clearly puts Acorn's promised £100 Prestel adaptor in the shade, though theirs is a direct-connect device with autodialling. Micronet too have one of these on the stocks.

I am particularly grateful to Mark Sealey and George Foot for their detailed comments about Micronet.

For further details, contact:
Micronet 800,
Durrant House,
8 Herbal Hill,
London, EC1R 5JB.

RESULTS OF BEEBUG'S THIRD COMPETITION

The response to our third software competition, which had a graphical theme, was very good and the entries were of an extremely varied nature. The main criterion for judging was visual content, with programming technique being taken into consideration.

We have awarded more prizes than was originally intended, and the final total comes to over £1300.

The first prize - a Luxor 14 inch portable TV/RGB colour monitor, was kindly donated by Portatel, of Sunbury-on-Thames, Middx. They specialise in producing monitors based on domestic TVs, and recommend the Luxor as being especially suitable for the BBC Micro. For further details of the Luxor see our review in BEEBUG Vol.1 No.3 P.13

We have awarded nineteen second prizes of the famous Wordwise word processing chip and 1.2 ROMs. Our thanks to Computer Concepts for donating some of these. Wordwise is a very powerful and easy to use word processor which is a great bonus to any BBC micro owner. It was reviewed in BEEBUG Vol.2 No.2 and is on special offer through BEEBUG.

Five Third prizes of £10 cash and the "Creative Graphics" book from Acornsoft have been awarded. Thanks to Acornsoft for donating these books, which are extremely interesting to anyone using graphics on the BBC Micro. The book was very favourably reviewed by BEEBUG in Vol.1 No.9.

A further twenty nine fourth prizes of £10 have also been awarded.

******* FIRST PRIZE WINNER *******

The winner of the first prize, a Luxor 14" TV/RGB was Mr. T. Loveland.

SECOND PRIZE WINNERS

The following have won prizes of a Wordwise wordprocessor and a 1.2 OS:

I. Sharp	B. Walker	G. Bains
B. Lintell	I. McAlpine	R. Snowdon
M. Banthorpe	A. Collyer	A. Phillips
P. Jackson	D. Clark	D. Hoskins
A. Dickinson	S. Crump	D. Chown
D. Cleverly	R. Weeks	M. Inglis
J. Webb.		

THIRD PRIZE WINNERS

The following have won the book "Creative Graphics" from Acornsoft plus £10:

H. Sugiura	A. Hynes	S. Wilkinson
J. Harris	Jon Watson	

CASH PRIZE WINNERS

The following have won a cash prize of £10:

C. Middlebrook	M. Tapley	M. Claxton
M. Bolderston	A. Harley	M. Diplock
K. Simpson	G. Cooper	P. Roden
G. Middleton	J. Kay	A. Farman
D. Johnson	R. Smith	W. Johnston
K. Southwood	R. Hoskin	R. deVekey
J. Jones	R. Meek	T. Hadley
Mrs. Sey	Lars Eber	A. Pratt
J. Davies	A. Morriss	G. Benson
A. Wood	H. Roerdinkholder	

Tested on O.S. 0·1 and 1·2 and on Basics I and II

DUAL SCREENS ON THE BEEB

by Colin Opie

By using the techniques described in this article you will be able to set up and use 'multiple' screens for the purposes of display and switch between them in a fraction of a second! With a bit of ingenuity such techniques can be used to create extremely versatile demonstration programs.

The version of the VDU23 statement which enables you to access the 6845 CRTC chip registers, (rather than redefine characters), has appeared in a number of past BEEBUG magazine articles:

a) "Cursor Delete" Vol.1, No.3, P.26
b) "Accessing the 6845" by John Yale, Vol.1 No.8 P.29
c) "Sideways Scrolling" (HINT), Vol.1 No.9 P.14
d) "Working Memory Display" by G. Greatrix, Vol.2 No.3 P.8

and is also documented in the User Guide (P.384).

One extremely useful facility which can obtained through this statement, and which article d) above utilises, is the re-mapping of screen display memory. In other words it is possible for you to define the area of memory which is to be used for a given screen mode display (except mode 7 - the teletext mode). This article explains how this can be achieved, using mode 5 as an example.

In what follows I will look at the theory behind the process. You may wish to move straight to the section entitled 'Running the program', and try out the dual screen demonstration program printed at the end of the article.

THEORY BEHIND PROGRAM

The 6845 CRTC chip is the device which controls the screen display when you are in modes Ø to 6. (It also has a part to play in mode 7 but its control is not so autonomous). Each of these modes uses up a certain amount of memory within your machine. For example, modes Ø to 2 use up 20k. When in one of these screen modes HIMEM is set to &3ØØØ and a Basic program may, on a cassette machine, use memory from &EØØ up to &2FFF. The VDU23 statement enables us to change the screen memory area, currently &3ØØØ to &7FFF to anything we like, though clearly it would not be terribly useful to assign the screen display area to 20k of pure ROM!

Coming back to the idea of multiple screen-page displays there is a practical constraint with regard to the amount of memory available to us. If we want to set up even two pages we would be in trouble in any of the modes except for modes 4, 5, and 6 - because modes Ø to 3 use between 16k and 20k each per screen, and therefore two screen-pages would require between 32k and 40k of memory when there is only 32k of user RAM on a standard model-B! As such we will consider mode 5 as an example of multiple paging, and set up two screen-pages. The program which performs this task is listed at the end of this article and I will make reference to it as I describe the theory of operation.

The first job is to inform the Basic system that HIMEM is not going to be what it thinks it should be for the chosen screen mode (see line 130). Normally HIMEM is set to &58ØØ and the screen memory area for mode 5 is therefore from &58ØØ up to &7FFF. To be on the safe side we have set HIMEM to &2FFF so that our second 'page' can reside in memory from &3ØØØ to &57FF. The two main points which we need to concern ourselves about are:

a) How can we write to our second page of screen memory?
b) How can we switch (rapidly) between the two 'pages'?

Let's consider first a method for writing into the second page.

Screen addressing, in conjunction with possible scrolling, can often be a little tedious. Secondly, the mapping

of 'characters' to 'character positions' in modes controlled by the 6845 CRTC is also fairly complex. It would therefore seem unwise to write relatively large chunks of program which would access screen-page two by 'poking'. A quicker and easier approach is to set up the contents of page two using the normal screen area (ie. page one), and then block move the whole 10k of memory into page two (line 160). When that is completed, page one can be created. This is in fact what the first part of the main program does (lines 140-180). Once both pages have been created we can set up a loop to enable us to have control over which screen-page is viewed (lines 200-240).

Now let's consider the PROCedure(s) for changing the page displayed. Two procedures are required in this case, one for each page, and in the example program these are called 'PROCpage1' and 'PROCpage2' (lines 260-340). They are identical in terms of 'theory of operation' and therefore only one need be discussed fully. Suppose we look at PROCpage2. Registers 12 and 13 of the 6845 CRTC hold the lowest memory address/8 that is to be used for the screen display in any given mode. Register 12 holds the high byte and register 13 the low byte. We write to these registers by using the form of the VDU23 statement:

VDU23,0,R,V,0;0;0;

where R is the register 'address' (in this case 12 or 13), and V is the value we wish to place into that register. For our 'page two' the lowest address used is &3000. The high byte of this is &30 which equals 48 decimal. Dividing this by 8 gives us the value 6. So, register 12 must have the value 6 put into it. The low byte of our address is zero and therefore register 13 must have zero written to it (see line 320). Coupled with this register change in the 6845 CRTC we also have to tell the 1.2 Operating System what the current 'high-byte' value of HIMEM is. This value is stored at location &34E, and for our 'page two' this will need to be programmed as &30. (This appears not to matter in OS 0.1). Using the above description of PROCpage2 you will be able to verify that PROCpage1 is correct for the 'standard' screen-page.

RUNNING THE PROGRAM

Type the program in and then save it on tape or disc. If you have typed it in correctly you will see the following happen: 'Page 2' will be created and consists of a block of yellow (or 'bright' if you using a black & white set) space invader characters. After a couple of seconds delay you will see 'Page 1' appear as a block of red (or 'dark') space invaders in a slightly different position (watch the legs and feelers!).

At this point the main loop of the program (lines 200-240) has been entered and typing a '2' will give you the 'Page 2' display. Typing a '1' will get you back to 'Page 1'. Notice how quickly the display changes! When you get tired of this little demonstration it is necessary to get back to 'Page 1' before pressing ESC, or simply press BREAK (followed by typing OLD if you want the program back).

A good demonstration of the speed of this screen-page change can be observed if you insert the line:

195 PROCpage1:PROCpage2:GOTO195

into the program. This stops the main loop of the program from being entered and merely continuously swaps the screen display between the two 'pages'. On a colour screen you will obtain something very close to (extremely 'frantic') orange space invaders!

CONCLUSION

The demonstration screens supplied in this article are extremely simple but it would be possible to create two quite complex graphics displays for whatever application you can think of. It would also be possible, for example, to have a disc library of 'page-2' screen displays and these could be loaded quite transparently by the suitable use of:

*LOAD picture

within the program, provided they had been saved by eg:

SAVE picture 3000 57FF

A little bit of experimentation in these areas could enable an extremely versatile demonstration/teaching package to be produced - happy programming!

```
  100 REM Multiple Screen Page Displays
  110 REM C.N.OPIE & A.WEBSTER - BEEBUG
  120 Version 1A
  130 MODE 5:HIMEM=&2FFF
  140 CLS:PRINT''"Page Two"
  150 PROCcreate_page_2
  160 FORI%=0 TO &27FC STEP 4:!(&3000+I
%)=!(&5800+I%):NEXTI%
  170 CLS:PRINT''"Page One"
  180 PROCcreate_page_1
  190
  200 REPEAT
  210 A$=GET$
  220 IF A$="1" THEN PROCpage1
  230 IF A$="2" THEN PROCpage2
  240 UNTIL FALSE
  250
  260 DEFPROCpage1
  270 VDU23,0,12,11,0;0;0;23,0,13,0,0;0
;0;0;
  280 ?&34E=&58
  290 ENDPROC
  300
  310 DEFPROCpage2
  320 VDU23,0,12,6,0;0;0;23,0,13,0,0;0;
0;0;
  330 ?&34E=&30
  340 ENDPROC
  350
  360 DEFPROCcreate_page_1
  370 COLOUR 1
  380 VDU23,225,192,153,60,90,126,60,66
,129
  390 FOR down%=6 TO 14 STEP 2
  400 FOR along%=2 TO 16 STEP 2
  410 VDU31,along%,down%,225
  420 NEXT along%,down%
  430 ENDPROC
  440
  450 DEFPROCcreate_page_2
  460 COLOUR 2
  470 VDU23,225,129,90,60,90,126,60,66,
36
  480 FOR down%=6 TO 14 STEP 2
  490 FOR along%=2 TO 16 STEP 2
  500 VDU31,along%,down%,225
  510 NEXT along%,down%
  520 ENDPROC
```

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

EQUS EQUB EQUW EQUD IN Basic II - Matthew Rapier

These four commands are used in the assembler to enter data bytes and strings simply into the machine code. This avoids the necessity to exit the assembler and use the ? and ! operators to generate blocks of data or look up tables e.g. ']:?P%=..:['. All these commands are used in this example:

```
  10 P%=&6000
  20 [
  30 EQUS "FRED IS HERE"
  40 EQUB 13
  50 EQUW 65535
  60 EQUD 123456789
  70 ]
  RUN

6000
6000 46 52 45
     44 20 49
     53 20 48
     45 52 45 EQUS"FRED IS HERE"
600C 0D       EQUB 13
600D FF FF    EQUW 65535
600F 15 CD 5B
     07       EQUD 123456789
```

From this it can be seen that EQUS inserts a string, EQUB a single byte, EQUW a word (two bytes), and EQUD a double word (four bytes).

KEY IN USE - Mark Hiseman

This is an undocumented error, with ERR=250. It occurs when an attempt is made to alter the *KEY definition of a key whilst the contents of the key are being read. For example:

```
10*KEY9,RUN|MLIST|M
RUN
(Press f9)
```

The message 'Key in use at line 10' appears because the program tried to redefine key 9 before the LIST on the end of key 9 had happened.

VISUAL EFFECTS USING THE 6845

Programs by W Lancaster & D Shread

Quite extraordinary video displays can be achieved by re-programming the video controller chip, and Win Lancaster and Dan Shread have written a number of small programs that show the sort of thing that can be achieved. We have picked two of them and described them below. After looking at these examples you may like to try out some ideas of your own - and that's where the fun starts!

Both programs are developed from an original article on the 6845 by John Yale (BEEBUG Vol.1 No.8 P.29). John Yale's article explained the format of the VDU23 command used to program the 6845, and also the purpose for each of the 18 internal registers.

EXAMPLE 1 - BOUNCE

In this program a 'ball' is being bounced to and fro' across the width of the screen while simultaneously being bobbed up and down.

The screen is first initialised to operate in mode 0 (lines 20-50), and then a circle is drawn (lines 70-140). At this point a loop is entered to perform the bouncing ball display. The variable 'FLAG' is used to change the horizontal direction of the ball. Two VDU statements produce the display (see lines 180-190). The 'VDU23;5...' command changes the vertical sync value for the 6845 and hence produces the vertical 'bounce' effect. The 'VDU23;13.....' command changes the least significant byte of the start address of screen memory. This means that the 6845 will use a slightly different block of memory for the display. An effect of this is that the screen can appear to 'shift', and in our example this is in fact what makes the ball go from side to side.

```
 10 REM BOUNCING BALL
 15 REM VERSION 1A
 20 MODE 0
 25 VDU19,1,3;0;0;0;
 30 VDU23,1,0;0;0;0;
 40 *FX9,0
 50 *TV255
 70 R=100
 90 VDU29,400;700;
100 MOVE 0,0
105 GCOL0,1
110 FOR X=0 TO 6.4 STEP PI/30
120 MOVE 0,0
130 PLOT 85,R*SIN(X),R*COS(X)
140 NEXT
150 A%=0
160 REPEAT
170 FOR DELAY=1 TO 50:NEXT
180 VDU23;5,A%;0;0;0;
190 VDU23;13,A%;0;0;0;
200 IF A%=254 FLAG=1
210 IF A%=0 FLAG=0
220 IF FLAG=1 A%=A%-2 ELSE A%=A%+2
400 UNTIL FALSE
500 END
```

EXAMPLE 2 - SWING

This program produces a 'swinging' circle going from left to right and back again. Each time it reverses direction it changes colour. It is novel in the sense that it enables you to apparently view the circle from different angles.

It works by first initialising the 6845 and the screen (lines 20-50) and then drawing a circle (lines 60-110). At this point the program enters a loop (lines 130-300) which produces the 'swinging' display. This effect is achieved by changing the number of characters per line (line 175).

As the program is in mode 0 there should be eighty characters per line. If this is changed to 79 the first character of the screen will remain exactly where it should be, but the last character will no longer fit onto the first line and so it is used as the

first character of the second line. Similarly the last 2 characters of the second line are put at the beginning of the third line, and so on. The visual effect of this is to 'slant' the display in one direction. Increasing the line length will therefore result in a slant the other way.

```
10 REM SWING
15 REM VERSION 1A
20 MODE 0
30 VDU23,1,0;0;0;0;
35 VDU23;2,106;0;0;0;
40 R=450
50 VDU29,640;512;
60 MOVE 0,0
70 MOVE 0,0
80 FOR X=0 TO 6.4 STEP PI/30
90 PLOT 85,R*COS(X),R*SIN(X)
100 MOVE 0,0
110 NEXT X
120 A%=64:FLAG=0
130 REPEAT
150 VDU19,1,4+FLAG*2;0;0;0;
160 FOR DELAY=1 TO 200:NEXT
175 VDU23;1,A%;0;0;0;
180 IF A%=106 FLAG=1
190 IF A%=64 FLAG=0
200 IF FLAG=1 A%=A%-1 ELSE A%=A%+1
300 UNTIL FALSE
```

BRAIN TEASER

by Colin Lindsay

This month we are giving a prize of £10. The organisation and judging will be carried out by Colin Lindsay and his decision is final.

THE COMPETITION

The last year whose digits formed a palindromic prime was 929 AD. When is the next such year?

(A palindrome is a number which reads the same backwards as forwards.
A prime number is a number which has no factors other than itself and unity.)

A prize of £10 will be awarded to the fastest Basic program that produces the next palindromic prime date after 929 AD. The program must be written entirely in BBC Basic (as a guide it should take less than 15 minutes to complete the task).

Entries must be sent on cassette and a suitable SAE enclosed for its return.

Send your entries to: C. Lindsay, 40, Halliwell Street, Chorley, PR7 2AL.
Entries sent to the wrong address, or entries arriving after 31/8/83 will be rejected.

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

LISTING PROGRAMS IN PAGE MODE - P.J.LeSueur, J.P. Jakubovics

When a program is listed with Ctrl-N selected it is very easy to forget to disengage page mode with Ctrl-O afterwards. This can prove extremely irritating when Shift has to be pressed to make a program function correctly. The solution is to define a key to do the listing, selecting page mode, listing and disengaging page mode afterwards. This only works if the program listing is allowed to terminate normally. If Escape is pressed the keyboard buffer is cleared so the Ctrl-O never happens. *FX230,1 stops the buffer being flushed so that a Ctrl-O on the end of the definition will happen. A suitable key defintion is:

```
*KEY0,*FX230,1|M|ML.|M|O*FX230,0|M
```

If *FX230,0 is not executed then Escape will not flush any buffers, so it will not switch all the sounds off, for instance.

MICROTAX REVIEWED

by Graham Greatrix

Program: MICROTAX
Supplier: Tax & Financial Planning Ltd, Barratt House, 7 Chertsey Road, Woking, Surrey, GU21 5AB
Price: £24.94 inc VAT
Reviewer: Graham Greatrix
Requirements: Model B, or 32k Model A (Not a disc system). Works on any OS.

This is a complete tax system. It contains a cassette holding about 220k of Basic in 15 programs. Supporting the cassette is a well-presented, readable and fully detailed UK income tax guide. The guide can be used on its own, and is similar to other guides available in book shops. Instructions for using the cassette on the BBC micro are contained on a card inside the library case. Several forms for recording data produced by the programs are also included in case the user has no printer. It is quite difficult, when reading an ordinary tax guide, to know just what is relevant to your particular circumstances. When your tax return is completed, you may still be unsure as to whether you are a tax evader or a tax avoider. (It is illegal to be a tax evader, and you will be financially poorer if you are not a tax avoider). What is needed, is a program that will take you step by step through every stage of filling in your tax return; that refers you to the relevant parts of the guide only when necessary; that keeps a constant check on your total tax liability, and that advises you of the most advantageous tactics. This system does just that.

The cassette programs loaded satisfactorily and are automatically chained. All the displays are in mode 7 and well laid out. Most programs are between 30 and 60 blocks in length and take quite a time to load. A disc version would be a distinct advantage here, but the cassette version supplied will not run directly if transferred to a disc. If you have a printer, the program will provide a print-out of all the important data.

The printer procedure consists of finding the ASCII character at each of 960 successive screen locations, and sending that to the printer. This produces rather too many blank lines in the print-out. There are no CHR$(13) characters on the screen, and so the procedure needs to insert these at the end of each screen line. Programs on Side A of the cassette did this satisfactorily, but not those on Side B. For example, line 4060 in the final program reads PRINT CHR$(A); whereas it should read:

```
VDU1,A:IF I MOD 40=39 THEN VDU 1,13
```

When the system bypasses programs that are not relevant for your tax return, it asks you to advance the tape to a particular value on the tape counter, however cassette counters differ widely, so this can only be a very rough guide. Unfortunately, the screen clears when the cassette motor is turned on, so that the tape counter value is lost from view. There were no other detectable errors or omissions which, considering the size of this package, is extremely good. Another point worth making is that it would be useful to use a colour TV/monitor because blue does not show up too well in black and white.

If you are an employee and have only one source of income, the package will take about two hours to work through. If you are a director of several companies, with fringe benefits, income from foreign and UK sources as well as from investments, you had better advise your secretary to allocate a whole day.

Altogether, Microtax is excellent for those with anything other than the simplest of tax returns. However, it is designed specifically for assisting with the 1983/4 return and for assessing your liability for 1982/3. You could need a completely new Microtax next year, and certainly whenever the taxation laws are changed. The one aspect that worries us is the need perhaps, to buy a new copy every year.

QUICK QUIZ RESULTS

by Sheridan Williams

In Vol.1, No.9, we published a "Quick Quiz". The problem was to write the quickest program that found all the integers between 0 and 999 that were the 'sum of the cubes of their digits'.

The following people submitted solutions that bettered my time of 2.98s (See Vol.2, No.1, P.6). Tony Cheal was the winner with a staggeringly fast 0.37 seconds, and he will be receiving a cheque for £10 from us.

Tony Cheal	of Cambridge	0.37	Andrew Cattell	of Loughborough	2.14
Andrew Kingwell	of Northampton	0.54	Julian Davey	of Cambridge	2.30
T.D. Reed	of Billingshurst	0.98	Ian Tresman	of Elstree	2.31
Dick Knowles	of Letchworth	1.17	P.D. Newth	of Stanley	2.33
Laurence Cox	of Beckenham	1.75	Mike Harrison	of Ealing	2.49
Paul Fairhurst	of Wigan	1.94	W. G. Smethurst	of London	2.54
W. Learmonth	of Sunderland	1.98	David Connolly	of Pinner	2.60
Peter Ellington	of Brighton	2.10	Jack Pike	of Chawston	2.94

The Winning Entry

```
    1  TIME=0
    2  A$=" "
    3  DIM A%(9),B%(9),C%(9*9*9-9)
    4 FOR A%=1 TO 9
    5   C%=A%*A%*A%-A%
    6   A%(A%)=A%*99-C%
    7   B%(A%)=A%*9-C%
    8   C%(C%)=A%
    9   NEXT
   10 FOR A%=0 TO 9
   11   D%=A%(A%)
   12   FOR B%=0 TO 9
   13     IF C%(ABS(D%+B%(B%))) PROCP
   14     NEXT,
   15   PRINT"That took ";TIME/100;"s"
   16   END
   17   DEF PROCP
   18   IF D%+B%(B%)<0 ENDPROC
   19   IF D%+B%(B%) PRINT
;A%;B%;C%(D%+B%(B%));A$; ELSE PRINT
;A%;B%;0;A$;A%;B%;1;A$;
   20   ENDPROC
```

Tony's method uses mathematical principles to construct a new algorithm.
You may like to consider the fact that:-

Solutions occur when:

$$A^3+B^3+C^3=100*A+10*B+C$$

This leads to:

$$(A^3-100*A)+(B^3-10*B)=(-C^3+C)$$

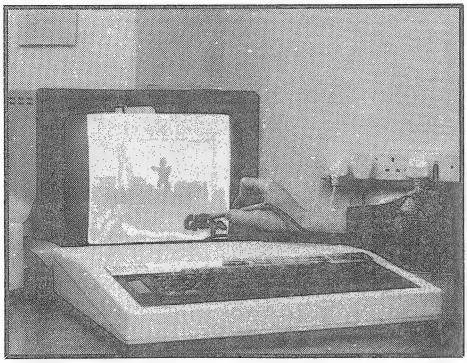
I would like to thank all those who entered. I am afraid that we cannot accept any further entries to this quiz.

Tested on O.S. 0.1 and 1.2 and on Basics I and II

BUILD YOURSELF A LIGHT PEN (32k)

by John Cole

Most people trying a light pen for the first time are delighted by the discovery. We are so used to TV screens being one-way mirrors that there is something magical about the ability to reach through in the opposite direction. Professional light pens are quite expensive but it is possible for even the beginner to make one himself at a fraction of the cost. This article explains how.



The BBC Model B has the provision for a light pen within the 'Analogue Input' interface, although the light pen signal is in fact a digital one! The particular pen described here is very simple mechanically, with all the parts fitting within the barrel of a suitably wide pen (at least 7mm internal diameter). A common feature of professional light pens is some form of switch, used to inform the computer of times when the pen signal is valid. This could readily be incorporated into our model although a neat mechanical construction is quite difficult to achieve. Our demonstration program compensates for the lack of a switch by suitable software.

HOW THE LIGHT PEN WORKS

At the most basic level the pen functions as a detector of light. The TV or monitor display is formed by a spot of light that traces out the lines making up the picture, left to right and top to bottom. In the Beeb this "raster-scan" process is controlled by the 6845 CRT controller chip (see BEEBUG Vol.1, No.8, P.29). Every time the spot passes the tip of the pen, the pen generates a digital pulse. This in turn causes the 6845 to store a number in two of its registers (R16 - high byte and R17 - low byte), dependent upon the RAM address that is currently being used to create the screen. The number has a minimum value when the pen is at the top left hand corner of the screen, and increases in unit steps to a maximum value at the bottom right. In any given display mode, the resolution is always much worse than the graphics resolution, but it may be better than one character position. For example in modes 0 to 2 the character resolution is 80x32.

All this is probably more than you need to know to use the pen, but certain consequences do have to be borne in mind to avoid some frustrated mystification:

1. The pen needs to detect light, so it works best with solid white on the screen and not at all with solid black.
2. When the pen is removed from the screen the 6845 retains the reading when the pen was last triggered.
3. Since the position is measured by pulse timing, an empirical adjustment needs to be made in software, not only for delays in the pen and the 6845, but also in the display device itself. For instance, a monitor using the RGB input may differ from a UHF colour TV by a few units of position.
4. Text scrolling changes the mapping of the RAM to the screen, so either begin afresh with a CLS or MODE statement, or subtract the start address in R12/13 from the contents of R16/17.

CONSTRUCTION

The circuit and layout are shown in Figures 1 and 2. Start by disassembling the pen and discarding all the parts except the barrel and the plastic plug on the end. Drill a hole in the plug so that the connecting flex is a tight fit. Saw or file 4mm off the writing

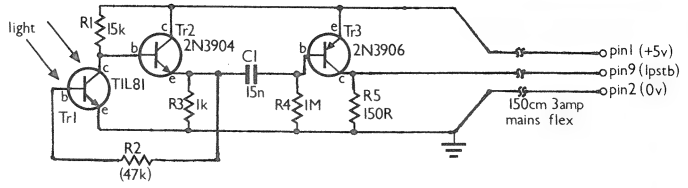


Figure.1 - All resistors are 1/4 or 1/8 watt carbon film. R2 should be varied to adjust sensitivity (see text). Tr1 (TIL81) may sensibly be replaced with a 'BPX25'.

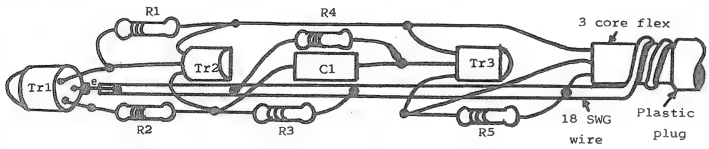


Figure.2 - The components are shown separated for clarity, but they should be soldered close to the 18 SWG wire with 1mm sleeving on each lead.

end of the barrel and enlarge the hole to 5mm to accept the phototransistor (TIL81). Stretch 20cm of 18 SWG copper wire in a vice to stiffen it, and wind one end twice around the flex to form a cable clamp touching the plastic plug. Cut the other end to a suitable length and solder it to the emitter lead of the TIL81 (see figure 2 to see which is the emitter lead). Clip off (or bend down) the small square tag next to the emitter lead. If all is well the TIL81 should reach neatly into the tip when the plug is replaced in the barrel. If this is not the case adjust the length of the wire and/or re-solder the TIL81. Now solder the D-connector to the far end of the flex and solder the rest of the components along the 18 SWG wire as shown in Fig 2, ensuring that the assembly will still slide inside the barrel. Finally, look critically for wires that could short together and keep them apart either by forming a bridge with quick-setting epoxy or by using insulating tape. (This is specially important for the +5 volt connection).

When the pen is finished, try it out with the demonstration program below. Ideally the pen should only work within 3cm of a white screen, though this is dependent on the brightness and persistance of the screen. If it seems too insensitive try turning up the brightness control or increasing the value of resistor R2. If on the other hand it is too sensitive, decrease the value of R2.

For those conversant with analogue circuitry the main point of interest is that the speed of response of the TIL81 phototransistor is improved by biasing it into conduction, and applying negative feedback to the base. If you can't get hold of a TIL81 for some reason then you could use a BPX25 phototransistor, as this works equally as well.

We obtained all the components from MAPLIN Electronic Supplies at the following address: PO Box 3, Rayleigh, Essex SS6 2BR.

Here is a full components list with the MAPLIN part number and price.

Part	Number	Price
R1	U15K	0.03
R2	U47K	0.03
R3	U1K	0.03
R4	U1M	0.03
R5	U150	0.03
C1	* see below	
TR1	QF30H	4.20
TR2	QR40T	0.17
TR3	QR42V	0.14
18 SWG Wire	BL46A	1.25
15 Pin Male D-connector	BK58N	1.45

*This capacitor should be as small as possible to fit inside pen. We used a 10n disc ceramic - order number YR73Q costing 5p.

DEMONSTRATION PROGRAM 'PAINTER'

As far as the software is concerned the pen simply indicates what part of the screen is being pointed to, but according to the imagination of the software writer the apparent effect can either be as dull as choosing a menu-item in a business program, or as colourful as the stroke of a child's paintbrush.

In our program you are presented with a white screen and a palette of colours. Run the pen from left to right across the appropriate part of the palette in order to acquire a colour, and then begin to paint anywhere on the rest of the screen. To terminate the use of a colour, hold the pen still for a moment and the palette will re-appear. The screen can be 'cleaned' by pressing any key whilst the palette is displayed.

The program is fun in itself and keeps children quiet for hours, but it also illustrates some techniques for making best use of the pen. For instance, the variable OS% in line 280 provides the adjustment for timing delays. Increase or decrease its value by a few units until a vertical line follows the pen directly without an offset to left or right. The short assembler routine is used to read the 6845 registers, because in Basic there is too great a chance that the high order byte will change before the low byte can be read.

For series 1 operating systems, an alternative will be to use *FX19 to wait for the start of a new frame when both registers must be static. The use of a simple digital filter (lines 160 to 200) means not only that a reasonably smooth line can be drawn despite the limited positional resolution, but also that even black lines already drawn can be overwritten. Perhaps the most important lesson of all is that, despite the absence of a switch on the pen, it can effectively be switched on and off in software, provided the program is planned with this requirement in mind.

You can change the resolution of the pen by redefining character 254 in line 100 to be a smaller circle. For example:

```
100 VDU5,23,254,0,0,28,28,28,0,0,0
```

Alternatively you could remove line 220 and replace line 210 with:

```
210 PLOT69,XM%,YM%
```

```
10 REM PAINTER by J.Cole. v1A
20 DIM P% 100
30 REM Assembler to read light pen r
egisters of 6845 CRT controller.
40 [.pen LDX#16:LDA#17:LDY#16
50 STX &FE00:LDX &FE01
60 STA &FE00:LDA &FE01
70 STY &FE00:CPX &FE01:BNE pen
80 RTS:]
90 MODE2:GCOL0,135:COLOUR135:CLS
100 VDU5,23,254,112,248,248,248,248,2
48,248,112
110 PROCPALETTE
120 PROCSTART
130 C%=0
140 REM Draw broad line until pen is
stationary.
150 REPEAT PROCPEN
160 IF XP%>64 DX%=XP%-XM% ELSE DX%=0
170 DY%=YP%-YM%
180 IF ABSDX%>600 THEN DX%=0
190 XM%=XM%+DX%DIV4
200 YM%=YM%+DY%DIV4
210 MOVE XM%,YM%
220 VDU254
230 IFABSDX%<20 AND ABSDY%<20 C%=C%+1
ELSE C%=0
240 UNTIL C%>25
250 GOTO110
260 REM End of main program.
265
270 DEF PROCPEN:REM Get pen coordinat
es in graphics units.
280 OS%=1542:REM Change value to comp
ensate for timing errors.
290 T%=(USR(pen)AND&FFFF)-OS%
300 XP%=16*(T% MOD 80)
310 YP%=32*(32-T% DIV 80)
320 ENDPROC
325
330 DEF PROCSTART:REM Wait for pen do
wn.
340 REPEAT TWAS%=T%
350 T%=USR(pen) AND &FFFF
360 UNTIL T%=TWAS%+80
370 PROCPEN
380 XM%=XP%:YM%=YP%
390 SOUND 1,-15,150,1
400 ENDPROC
405
410 DEF PROCPALETTE:REM Draw palette
and wait for colour choice.
420 VDU4,23,255,&FCFC;&FCFC;&FCFC;&FC
FC;23;8202;0;0;0;28,0,31,0,0
430 COLOUR135:CLS
440 FOR T=128 TO 134
450 COLOUR T
```

```
  460 VDU32,255,255,32
  470 NEXT
  480 *FX15,0
  490 REPEAT PROCPEN:K=NOT INKEY(0):UNT
IL(XP%>16 AND XP%<64) OR K
  500 IF K CLG:GOTO430
  510 SOUND 1,-15,150,1
  520 COL=7-YP%DIV128
  530 GCOL0,COL
  540 COLOUR COL+128:CLS
  550 T=INKEY(100)
  560 COLOUR128:CLS
  570 VDU5
  580 ENDPROC
```

POINTS ARISING

DISC PROGRAM RELOCATOR - Vol.2 No.2

Several members have written to say that the "Disc Program Relocator" given in BEEBUG Vol.2 No.2 P.31 does not work on Issue II Basic because you cannot have "ON ERROR DELETE" in Basic II. We apologise for this, and are now testing all programs on Basics I and II. The following routine works on both Basics.

To use the routine simply add these lines to the beginning of the program to be moved down and CHAIN the program from disc as normal.

```
1 *KEY0 *TAPE|MDELETE1,3|MFORI%=0TO
  TOP-PAGE STEP4:I%!&E00=I%!&1900:
  NEXT|MPAGE=&E00|MEND|MRUN|M
2 *FX 138,0,128
3 END
```

For further information please refer to the article "Disc Program Relocator" given in BEEBUG Vol.2 No.2 P.31.

HEDGEHOG - Vol.2 No.2

There was a minor problem with the listing of 'Hedgehog' in BEEBUG Vol.2 No.2 p.38. In line 880 someone had stolen the hyphen from PROC_HARDER. This was lost in our pasting up process, but fortunately most members realised that it was missing.

Secondly, there was an error in line 2460 which takes effect when you reach the fifth screen. The barriers in the middle overlap and give a 'String too long' error. This is cured by:

2460 LEVEL%=LEVEL%-2*(LEVEL%<>9)

In our tests of the program we never got further than screen 4. Thanks and congratulations to M.Gidley, R.Gallimore, and others for getting that far, and for bringing this to our attention.

DISC SECTOR EDITOR - Vol.2 No.3

Last month's magazine contained an article about a 'Disc Sector Editor' (Vol.2, No.3, P.27), which could be used to retrieve an accidently erased program or data file. The sector editor works well, but a problem has been found to arise if the retrieved program was not originally the 'first' one on the disc. If the disc is subsequently written to (eg. through SAVE or *COMPACT) then the DFS still thinks there are spaces on the disc where in fact there are not - and a file may get overwritten! The solution is simple and merely involves performing the following operations AFTER retrieving a file as described by the article:

1) *COPY the retrieved file onto another disc,
2) *DELETE the original retrieved file,
3) *COPY the file created by 1) back onto the original disc.

We have found no problems arising at all after performing these few extra steps.

FAST VERIFY FOR WORDWISE - Vol.2 No.3

The hint which appeared in last month's magazine on page 34 should not have the & before the hex value of 8000. This is because the statement *LOAD assumes that the value is in hex.

WATCHING THE BEEB AT WORK - Vol.2 No.3

The program printed on page 8 of last month's magazine does not work on OS 0.1. Our note to this effect became detatched from the copy. We hope that the absence of any note about which OS it was tested on may have made you suspicious (in spite of our note on page 2!)

PUTTING CASSETTE PROGRAMS TO DISC

by Alan Wood

The first thing one usually wants to do after installing a disc system, is to transfer all one's programs from cassette onto disc. If the programs are written in Basic there is often no problem. Sometimes the program doesn't fit because the disc system has reserved some of the user RAM for its own use, there are numerous routines to shift the program down to &EØØ once it has been loaded. (See "Points Arising" for our Program Relocator). So wouldn't it be great to transfer all those machine-code programs. This program is a version of the excellent "Disc Menu" program which appeared in BEEBUG (Vol.2 No.1), but which now permits almost all types of 'awkward' programs - Basic and machine-code - to run successfully on a disc system.

WHY DON'T THEY RUN?

There are many variations of 'awkward' programs, the main types are:

1. Single machine-code programs - these won't run because they start at &EØØ and the disc system reserves that space for its own use.

2. Programs divided into two or more programs - first a Basic part which displays the manufacturers advertising page, sometimes sets up things such as sound envelopes, and then *RUNs a machine code main program.

3. Basic programs which use high resolution graphics and often leave insufficient room when the disc system has taken its chunk of RAM.

This 'MENU' program will handle all these types and their variations as well as the 'normal' files which will run anyway.

THE 'MENU' PROGRAM

The program will accommodate up to 19 files and permits selection of any program with the press of a single key. The procedure 'move' assembles a machine-code routine into the top of RAM, out of harm's way, ready to shift a program down when necessary. Procedure 'select' takes the selected program and jumps to its own individual routine which handles all details of loading and running the selected program. There is no general purpose routine, and in this way almost any type of program can be dealt with.

GETTING IT ONTO DISC

There are several methods of copying from cassette to disc, some of the fundamentals are dealt with in the article "Secrets of Program Saving" (BEEBUG Vol.1 No.2 P.18).

The three main types are given below:

TYPE ONE

Single machine-code programs (eg. SWOOP).
To transfer this type, perform the following;

```
*TAPE       select the tape filing system
*OPT1,2     print the file information
*LOAD"" 1900 load the program from tape
```

As the tape loads you will see a message of the form:

```
 FILENAME  05 0585   FFFF1111 FFFFeeee
```

Now note down the file length (denoted by 1111) and the execution address (denoted by eeee) and continue with:

```
*DISC       select the disc filing system
*SAVE"FILENAME" 1900 +1111 7000
        which saves the program on disc.
```

Note that the execution address has been set to 7000 (in hex), this is where the machine-code 'move' routine is stored. When the file is *RUN it will jump to the 'move' routine which will automatically shift the file down to &EØØ and then jump to the actual execution address noted earlier.

TYPE TWO

Two-file programs (eg. MONSTERS, SNAPPER etc.).

These use a Basic program to display the advertising and to *RUN the main machine-code section.

The Basic section is odd because, if you LOAD it then SAVE it, the SAVEd version will inexplicably be much shorter than the original! (and will not run). This is because the graphics data is stored in the RAM space immediately following the Basic part, and is thus not a part of the Basic program.
To copy this type do the following:

```
*TAPE
*OPT1,2
LOAD""      (NB do not use *LOAD)
```

As the tape loads you will see a message of the form:

```
FILENAME  Ø5 Ø585   FFFF1111 FFFFeeee
```

Now note down the file length (denoted by 1111) and the execution address (denoted by eeee) and continue with:

```
*DISC
*SAVE"filename" 19ØØ +1111
```

Note the use of *SAVE, this is to ensure that the graphics codes are saved as well as the Basic.
LOAD was used so that the program can be LISTed. LIST the program and find the line which *RUNs the main machine-code section. This line may not be acceptable to the disc system for one of the following reasons:

a) The program name should contain no more than seven characters.
b) The program name must not be the same as the Basic section's program name, otherwise the disc system will be confused.
c) There may be no program name at all (eg. *RUN""), this is not acceptable to the disc system and a program name must be chosen.

If changes to this line were necessary then re *SAVE it as above.

TYPE THREE

BASIC programs are transferred to disc in the usual way;

```
*TAPE
*LOAD""
*DISC
*SAVE"filename"
```

Do not attempt to run any programs during transfer, and make sure that the filename is no more than 7 characters long.

CUSTOMISING THE 'MENU' PROGRAM

It is now necessary to put information about each file into the 'menu' program and to write the individual routines which will enable each file to run. Proceed as follows:

1. Choose a title for the menu and type it into the DATA statement at line 47Ø.
2. Starting at line 48Ø enter data about each file to be included. The first item is the file name, followed by the program name to be displayed by the menu, followed by a short description of the program.
3. Starting at line 3ØØ enter a routine for each program in turn which will set it up and run it.

Examples of these routines are as follows:

a) Machine-code programs - these must be shifted down to &EØØ before being run. First set the execution address (denoted by eeee when taking the file off the cassette) as the parameter of the procedure 'move', and finally *RUN the program.
The routine will look like this:

```
3ØØ PROCmove(&18ØØ):*RUN"SWOOP"
```

(Don't omit the & before the address).

b) Two-file programs - as above there is a machine code section so the execution address must be set up and the 'move' routine assembled into RAM. Since the first file is in BASIC it must be CHAINed. The line will look like this:

```
59Ø PROCmove(&3283):CHAIN"SNAPPER"
```

c) BASIC programs - which won't fit into memory, will almost always fit if PAGE is set to &11ØØ. Since the disc system normally only uses &EØØ to &11ØØ all that is required is to set PAGE to &11ØØ and CHAIN the program. The line will look like this:

```
6ØØ PAGE=&11ØØ:CHAIN"SPHINX"
```

On some Acornsoft programs the Basic section does nothing more than display

the advertiser's screen and *RUN the actual program, in this case the complete Basic section can be omitted if required and the game itself treated as a single machine code program.
Sometimes however there is more to it than that, the program will often do things like set up sound envelopes, or print instructions. If the extra instructions are short then they can be incorporated into the appropriate 'menu' routine and the Basic file deleted as above, otherwise they are best left intact.

4. Modify line 290 to include the first line number of each separate routine written, this ensures that after selection the appropriate action is taken for each file.

Other programs can be added later up to a maximum of 19.
The MENU can be run automatically by pressing SHIFT followed by BREAK, the instructions on how to do this are in the original 'Disc Menu Program' (Vol.2 No.1 P.28).

After all this there are still a very few programs which won't transfer on to disc without modification to the programs themselves. Examples are Philosopher's Quest and EDG Graphics system. These programs call machine code data into a given area of memory WITHOUT corrupting the original Basic program, when this is done with a disc system the area &E00 to &1100 is corrupted.

One can only hope that the software distributors will do something for disc owners, it would be a nice gesture if they would offer a 'trade-in' allowance to those who bought it on cassette only to discover too late that it won't run on disc, or perhaps publish program modifications to do it oneself.

```
  1 REM Cassette to Disc Version 1A
 10 ON ERROR GOTO 560
 20 READ title$
 30 DIM F$(20),N$(20),D$(20)
 40 I%=0
 50 REPEAT
 60 I%=I%+1
 70 READ F$(I%),N$(I%),D$(I%)
 80 UNTIL F$(I%)="DONE"
 90 max%=I%-1
100 MODE 7
110 PROCcentre(title$)
120 FOR J%=1 TO max%
130 IF max%<10 THEN PRINT
140 letter$=CHR$(64+J%)+"."
150 colour$=CHR$(129+(J%-1)MOD 6)
160 PRINT letter$;colour$;N$(J%);TAB(
14);"- ";D$(J%)
170 NEXT
180 VDU 28,0,24,39,VPOS
190 PRINT'"enter the letter of your c
hoice....";
200 REPEAT
210 A%=GET
220 PRINT CHR$(A%)
230 req%=A%-64
240 IF req%>0 AND req%<=max% THEN PRO
Cselect
250 PRINT'"select a letter from A to
";CHR$(max%+64);"....";
260 UNTIL FALSE
270 END
280 DEF PROCselect
290 ON req% GOTO 300,310,340,350,360,
370,380
300 PROCmove(&1800):*RUN"SWOOP"
310 ENVELOPE1,4,6,-3,-3,4,2,2,0,0,-1,
0,63,58
320 ENVELOPE2,3,-4,-1,2,6,6,28,81,-4,
-5,-1,126,63
330 PROCmove(&3283):*RUN"SNAPPER"
340 PAGE=&1100:CHAIN"SPHINX"
350 PROCmove(&E02):CHAIN"DEFEND"
360 PROCmove(&E02):CHAIN"MONSTER"
370 PAGE=&1100:CHAIN"FROGLET"
380 PAGE=&1100:CHAIN"DEATHW"
390 ENDPROC
400 DEF PROCcentre(T$)
410 PRINT'
420 L%=LEN(T$)/2
430 FOR Y%=1 TO 2
440 PRINT CHR$141;CHR$134;TAB(20-L%);
T$
450 NEXT
460 ENDPROC
470 DATA GAMES DISC 1
480 DATA SWOOP,SWOOP,Space invaders g
ame
490 DATA SNAPPER,SNAPPER,Chase throug
h a maze
500 DATA SPHINX,SPHINX,Adventure game
510 DATA DEFEND,DEFEND,Space invaders
game
520 DATA MONSTER,MONSTER,Chase game
530 DATA FROGLET,FROGLET,Cross to saf
ety
540 DATA DEATHW,DEATHWATCH,Battlefiel
d shooting
550 DATA DONE,DONE,DONE
560 REPORT
```

```
570 IF ERR<200 PRINT ERL:END
580 PRINT"  ( ";F$(A%-64);" )"
590 X=INKEY(500)
600 RUN
610 END
620 REM assemble the 'move' routine
630 REM into the top of RAM.
640 REM The routine is called by the
650 REM file needing to be moved and
660 REM then jumps to the file's
670 REM execution address to run it.
680
690 DEF PROCmove(execute%)
700 base=&70:old=&BE:length=&C2
710 FORI%=0 TO 2STEP2:P%=&7000
720 [OPTI%
730 LDA #0
740 STA base
750 LDA #&0E
760 STA base+1
770 LDY #0
780 LDX length+1
790 BEQ LOLOOP
800 .LOOP1
810 LDA (old),Y
820 STA (base),Y
830 INY
840 BNE LOOP1
850 INC old+1
860 INC base+1
870 DEX
880 BNE LOOP1
890 .LOLOOP
900 LDX length
910 BEQ FINISH
920 .LOOP2
930 LDA (old),Y
940 STA (base),Y
950 INY
960 DEX
970 BNE LOOP2
980 .FINISH
990 LDA #&0E
1000 STA &18
1010 LDX #&FF
1020 TXS
1030 JMP execute%:]NEXT
1040 ENDPROC
```

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

*FX202 - Dr.R.W. Harris

The *FX202 call returns the status of the keyboard LEDs and the shift and control keys. Used as an OSBYTE call with X=0,Y=255 it returns the following in X:

keys pressed	caps lock on	shift lock on	neither
shift	&28	&18	&38
control	&60	&50	&70
shift/control	&68	&58	&78
neither	&20	&10	&30

(For more on FX202 see "FX Call Update" in BEEBUG Vol.2 No.2 P.28)

HOW TO "NEW" A RUNNING PROGRAM (New O.S.) - Tom Standage

(As demonstrated in BEEBUG Vol.1 No.10 P.16)
Simply insert the lines:

```
1000 *KEY0,NEW|M
1010 *FX138,0,128
1020 END
```

This will insert the word 'NEW' into the keyboard buffer, followed by a return. Thus the program will be erased after it stops running. It is of course still possible to OLD the program after this. To make the program un 'oldable' replace line 1020 with:

```
1020 !PAGE=0:!(PAGE+4)=0:END
```

STRING SEARCH IN WORDWISE

Using option 5 of Wordwise for a selective search is a very fast way of finding an occurrence of a word. Pressing Escape when it prompts for the replacement will leave the cursor on the word.

USING FILES (PART 5)

by Sheridan Williams

This is the last part in our series on using files. To conclude I will discuss two topics which will be of interest to those who want to use files with more complex design - Addressing Algorithms, and Index Sequential files.

This article is necessarily complicated and should be read in conjunction with 'Files Part 4' as the topic of 'Direct Access Files' was discussed there.

ADDRESSING ALGORITHMS

Because random access files require fixed length records, we need to have a method for finding the start address for each record. Calculating this start address is done by using the key field. The key field is the field that uniquely defines a record, it must be different for every record, so that there is no ambiguity.

The key field should lend itself to address calculation, for example: Part number 1051, Clock number 117, Credit card number 0704552345.

For example in a stock file where the record length is 50 characters, then:

part number	address
1	1
2	51
3	101
.	.
p	50*(p-1)+1

Therefore part number 1051 would start at address 50*(1051-1)+1 = 52501

However, part numbers might consist of four digits starting at 1000 and by using the above algorithm 999 records would be wasted. A better algorithm in this case would be 50*(p-1000)+1. Matters become more complicated if part numbers are not continuous, they might exist only in the following ranges: 1000-1999 and 3000-6999. In this case a suitable addressing algorithm would be:-

```
50*((p-1000)+(p>1999)*1000))
```

(See the article "Logic Part 2" BEEBUG Vol.1 No.6 for an explanation of logic within arithmetic statements).

It is also possible to calculate a start address for a record with a non-numeric key. Let us assume that the key field is alphabetic.

We can calculate an address based on the numeric equivalent of the letters. For example assuming that the record length is 20 characters the following will apply:

Key	Start Address
AAA	1
AAB	21
AAC	41
.	.
.	.
key$	26*26*FNa(1)+26*FNa(2)+FNa(3)+1

where

```
DEF FNa(x)=ASC(MID$(key$,x,1))-65
```

You will notice that the key ZZZ would require an address of over 350,000 which exceeds the capacity of one side of a normal (current) floppy disc drive.

INDEX SEQUENTIAL FILES

Index sequential access is the process of storing or retrieving data directly, but only after reading an index to locate the address of that item.

Index sequential files provide a solution to two problems:-

1. There is no need for a record to be of fixed length.
2. The record key can consist of any characters.

Briefly, each file actually consists of TWO files - the MAIN file and the INDEX file. I would recommend that the "index" file should have the same name as the "main" file but should have a

catalogue starting with "I"; for example if we wish to have a file called STOCKS and process it as an index sequential file, then the main file should be called STOCKS and its associated index file called I.STOCKS.

In order to access (read or write) any record, the index is consulted first. The index file has the briefest possible record structure, which simply contains the key and the address of the corresponding record on the master file.

KEY	DESCRIPTION
P1Ø23	Door handle (offside)
QN19zDX	Wheel nut
11XXVB	13ØØcc cylinder head

INDEX FILE	MAIN FILE
P1Ø23,1	Door handle (offside)
QN19zDX,24	Wheel nut
11XXVB,35	3ØØcc cylinder head

DEMONSTRATION PROGRAM

This program demonstrates the action of an Index Sequential file. The main file is called DEMO, and its index is called I.DEMO. Four routines are implemented - write a record, read a record, list the records, and end. Many other routines could be implemented, for example deletion of records, list specific records, printer on/off etc.

To use the program, first make sure that there are no files on the disc called DEMO and I.DEMO. Also make sure that there is room on the disc for two more files. Type RUN and select choice number 1 - "write a record". You will be asked for a "key", try entering ABC123V, then you will be asked for a record description, try entering: Ford Escort XR3i. This will then be written to the file and then the menu will re-appear.

Before you put on any more records you should try each of the other options to check if you have typed the program correctly (you should have no problem if you have read it from the magazine cassette). If all goes well you can enter lots more records, together with their keys, and retrieve them at will.

IMPORTANT NOTE FOR USERS WITH BASIC II

If you have Basic II you will have to replace the OPENIN with OPENUP in both lines 2Ø and 6Ø. If you have loaded the program from the "magazine cassette" then you will have no problem, because if you have Basic II the word OPENIN will be converted automatically to OPENUP. See the "Hint and Tip" elsewhere in this issue for further details.
This note also applies to the program given in last month's "Using Files - Part 4" article. Change the OPENIN to OPENUP in the first demonstration program given on page 21 line 2Ø, and in the program on page 22 line 3Ø.
[To discover whether you have Basic I or Basic II, switch on the machine from cold, now type REPORT. If the date in the message is 1981 then you have Basic I, if it is 1982 then you have Basic II. For further details on "Issue II Basic" see BEEBUG Vol.1 No.6 P.11 Item 7.]

```
    5 REM VERSION 1A
   1Ø  ON ERROR GOTO 7ØØØ
   2Ø  c1=OPENIN"DEMO":IF c1<>Ø THEN 6Ø
   3Ø
c1=OPENOUT"DEMO":c2=OPENOUT"I.DEMO"
   4Ø nia%=11:nma%=1:PTR#c2=1:
PRINT#c2,nia%,nma%
   5Ø  CLOSE#Ø:GOTO 2Ø
   6Ø  c2=OPENIN"I.DEMO"
   8Ø  REPEAT:CLS
   9Ø    PRINT TAB(7);CHR$13Ø;"INDEX
SEQUENTIAL FILE DEMO"
  1ØØ    PRINT''''1. Write a record"
  11Ø    PRINT"2. Read a record"
  12Ø    PRINT"3. List of records"
  125    PRINT"4. End"
  13Ø    PRINT''"Choice? ";:choice$=GET$
  14Ø    PRINT choice$
  15Ø    IF choice$="1" PROCwrite
  16Ø    IF choice$="2" PROCread
  17Ø    IF choice$="3" PROClist
  18Ø    IF choice$="4" CLOSE#Ø:END
  19Ø    UNTIL FALSE
  999 :
 1ØØØ  DEF PROCwrite
 1Ø2Ø  INPUT"Record key ",key$
 1Ø3Ø  INPUT"Record description ",desc$
 1Ø35  PROCread_header
 1Ø4Ø  PTR#c2=nia%:PRINT#c2,key$,nma%
 1Ø5Ø  PTR#c1=nma%:PRINT#c1,desc$
 1Ø6Ø  nma%=nma%+LEN(desc$)+2
 1Ø7Ø  nia%=nia%+LEN(key$)+7
 1Ø8Ø  PTR#c2=1:PRINT#c2,nia%,nma%
```

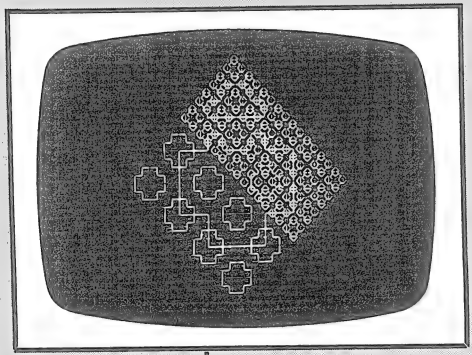
```
 1090  ENDPROC
 1999 :
 2000  DEF PROCread:LOCAL k$
 2020  INPUT"Key ",k$
 2025  PROCread_header:IF NOT exist
PROCready:ENDPROC
 2030  REPEAT
 2040    INPUT#c2,key$,nma%
 2050    UNTIL EOF#c2 OR k$=key$
 2060  IF k$<>key$ PRINT"Key not found
on file.":PROCready:ENDPROC
 2070  PTR#c1=nma%:INPUT#c1,desc$
 2080  PRINT"Key = ";key$
 2090  PRINT"Description = ";desc$
 2100  PROCready:ENDPROC
 2999 :
 4000  DEF PROClist
 4010  PROCread_header:IF NOT exist
PROCready:ENDPROC
 4016  REPEAT
 4020    INPUT#c2,key$,nma%
 4030    PTR#c1=nma%:INPUT#c1,desc$
 4040    PRINT"Key = ";key$
 4050    PRINT"Description = ";desc$'
 4060    UNTIL EOF#c2
 4070  PROCready:ENDPROC
 4999 :
 5000  DEF PROCready
 5010  PRINT"Press 'space' to return to
menu ";
 5020  REPEAT UNTIL GET=32
 5030  ENDPROC
 5999 :
 6000  DEF PROCread_header
 6010  PTR#c2=1:INPUT#c2,nia%,nma%
 6020  exist=(nia%>11)
 6030  ENDPROC
 6999 :
 7000  IF ERR=157 CLOSE#0:RUN:REM File
left unclosed
 7010  IF ERR<>17 PRINT:REPORT:PRINT"
at line ";ERL:END
 7020  IF ERL=130 PRINT''CHR$129;"You
must end using option 4":q=INKEY(200)
 7030  GOTO 80
```

SPIDER'S WEB (16/32k)

by J. Harris

This is a graphics program which draws intricate symmetrical shapes on the screen in different colours and sizes. It runs on any machine, but for the 16k machine, change line 90 to MODE 5.

The program runs quickly because of the rows of DRAW commands in lines 250 and 280. The computer requests a choice of two patterns, press ESCAPE to return to this menu. To 'freeze' the display you can press 'F', and to continue the animation press 'A' (this is handled by line 230).



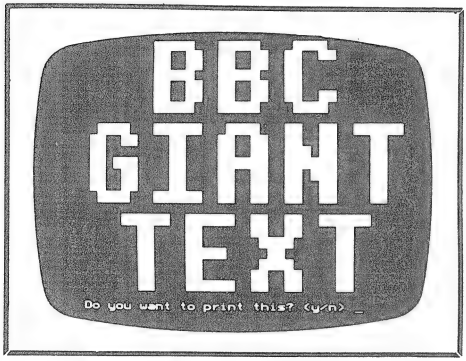
```
10 REM VERSION 1A
20 ON ERROR GOTO 300
30 MODE7
40 REPEAT
50 INPUT'"Pattern 1
or 2  ? "T$
60 T%=VAL(T$)
70 CLS
80 UNTIL T%=1 OR T%=2
90 MODE1:REM OR MODE 5
100 VDU 5
110 REPEAT
120 M%=400
130 REPEAT
140 VDU19,RND(3),RND(7);
0;18,3,RND(3)
150 GCOL3,RND(3)
160 PROCcross(640,512,512)
170 M%=M%/2
180 UNTIL M%<16
190 UNTIL FALSE
200 END
210 DEFPROCcross(X%,Y%,S%)
220 IF S%<M% ENDPROC
230 IFINKEY-68 REPEAT:UNTIL GET$="A"
240 S%=S%/2
250 IF T%=1 PROCcross(X%,Y%+S%,S%):PROCcr
oss(X%+S%,Y%,S%):PROCcross(X%,Y%-S%,S%):PRO
Ccross(X%-S%,Y%,S%) ELSE PROCcross(X%-S%,Y%
+S%,S%):PROCcross(X%+S%,Y%+S%,S%):PROCcross
(X%-S%,Y%-S%,S%):PROCcross(X%+S%,Y%-S%,S%)
260 VDU29,X%;Y%;
270 MOVE0,S%:U%=S%/2
280 DRAW U%,S%:DRAW U%,U%:DRAW S%,U%:DRAW
S%,0:DRAW S%,-U%:DRAW U%,-U%:DRAW U%,-S%:D
RAW -U%,-S%:DRAW -U%,-U%:DRAW -S%,-U%:DRAW
-S%,U%:DRAW -U%,U%:DRAW -U%,S%:DRAW 0,S%
290 ENDPROC
300 ON ERROR OFF
310 IF ERR=17 RUN
320 MODE7:REPORT:PRINT" at line ";ERL
330 END
```

Program tested on O.S. 0·1 and 1·2

GIANT SCREEN CHARACTERS (16/32k)

by Tim Powys-Lybbe

Have you ever wanted to display really BIG letters on the mode 7 screen. Well, this program does it, in fact it prints blown up mode 0 to 6 characters onto a Teletext mode 7 screen. This is possible as the structure of these characters can be found through the Operating System call OSWORD with A=10; line 310 does this. Line 330 then deciphers the pixels and prints them as big graphics blobs on the screen.



The program uses the Teletext colours, and a separate colour can be specified for each of the three lines on the screen. In addition, a rainbow effect can be specified for any or all lines, this gives a different colour for each mode 7 line on the screen. At the end of the program these colours will continually change to give a striking display, you exit this by holding down the space bar. The program was designed to give a large scale, attractive display.

Obviously the program can be modified to suit different purposes. For instance one could give a continuous series of screens each with a different message by READing from DATA into the screen display holding array, D%(2,14) and into K% and R%(2). The easy way to do this is to read the data into PROCLineInput and into PROCColourInput instead of using FNLineIn for keyboard entry. K% needs to be given the number of characters on each screen less one.

```
10 REM BIG characters Version 1A
20 MODE7
30 DIMD%(2,14),R%(2):B=32:REM D% hol
ds the characters for the display; R% h
olds the colours for the display.
40 PROCBigLetters
50 MODE7:END
60
70 DEF PROCBigLetters
80 VDU26,12,14
90 PRINT"   You can display a maximu
m of three"'"lines of five characters e
ach on the"'"screen."'
100 PRINT"   The characters on each l
ine will be"'"centralised on that line.
 However"'"spaces can be included to f
orce the"'"position on any line."'
110 PRINT"   The colour codes are:"'"
1 Red";TAB(13)"4 Blue";TAB(26)"7 White"
'"2 Green";TAB(13)"5 Magenta";TAB(26)"8
 Rainbow"'"3 Yellow";TAB(13)"6 Cyan"'"
   Enter your characters now:"
120 REPEATK%=0:FORI%=0TO2:PRINTTAB(0,
17+I%*2)"Line ";I%+1;SPC8;TAB(9,17+I%*2
);:PROCLineInput:PRINTTAB(19,17+I%*2)"C
olour    ";TAB(27,17+I%*2);:PROCColourI
nput:NEXT
130 PRINTTAB(0,24)"   Are your entrie
s OK? (y/n)  ";TAB(30,24);:PROCYesNo:UN
TILA$="Y"
140 PROCBigDisplay:ENDPROC
150
160 DEF PROCLineInput
170 A$=FNLineIn
180 IFA$="" ENDPROC
190 FORJ%=0TOLEN(A$)-1:D%(0,K%)=ASC(M
ID$(A$,J%+1,1)):D%(1,K%)=20-4*LEN(A$)+J
%*8:D%(2,K%)=I%*8:K%=K%+1:NEXT
200 ENDPROC
210
220 DEF FNLineIn
230 x=POS:y=VPOS:PRINTTAB(x,y)SPC8;TA
B(x,y);
240 B$="":REPEATA$=GET$:IFASC(A$)>31
ANDASC(A$)<127 ANDLEN(B$)<5 B$=B$+A$:PR
INTTAB(x+LEN(B$)-1,y)A$;
250 IFA$=CHR$(127)ANDLEN(B$)>1  B$=LE
FT$(B$,LEN(B$)-1):PRINTTAB(x,y)B$;" ";T
AB(x+LEN(B$),y); ELSEIFA$=CHR$(127)ANDL
EN(B$)=1 B$="":PRINTTAB(x,y)" ";TAB(x,y );
260 UNTILA$=CHR$(13)
270 VDU31,x,y:=B$
280
290 DEF PROCBigDisplay
300 IFK%=0:PROCBigLetters ELSECLS:C=0
:PROCColours:FORL%=0TOK%-1
```

```
  310 A%=10:X%=&70:Y%=0:?X%=D%(0,L%):CA
LL&FFF1
  320 FORM%=0TO7:FORN%=7TO0STEP-1
  330 IF(?(&71+M%)AND2^N%)=2^N% VDU31,D
%(1,L%)+7-N%,D%(2,L%)+M%,255
  340 NEXT:NEXT:NEXT
  350 PRINTTAB(0,24)"Do you want anothe
r screen? (y/n) ";:PROCYesNo:IFA$="Y" R
UN ELSEPRINTTAB(0,24)STRING$(38," ");:V
DU23;8202;0;0;0;:C=20:REPEATPROCColours
:UNTILB=32:ENDPROC
  360
  370 DEF PROCColourInput
  380 REPEAT:R%(I%)=VAL(FNLineIn):UNTIL
R%(I%)>0ANDR%(I%)<9:ENDPROC
  390
  400 DEF PROCColours
  410 FORI%=0TO2:FORJ%=0TO7:IFR%(I%)=8
B=INKEY(C):VDU31,0,I%*8+J%,144+RND(7) E
LSEIFC=0 VDU31,0,I%*8+J%,144+R%(I%)
  420 NEXT:NEXT:ENDPROC
  430
  440 DEF PROCYesNo:REPEATA$=CHR$(GETAN
D&DF):UNTILINSTR("YN",A$)>0:PRINTA$;:EN
DPROC
```

CONTACT POINTS FOR THE BEEB

A directory of addresses and telephone numbers
compiled by David Graham

MACHINE ORDERING

Order machines and accessories from any appointed dealer, or from:

Vector Marketing
Denington Estate
Wellingborough
Northamptonshire
NN8 2RL
[Tel 0933 79300]

It is always advisable to check supply before ordering.

SERVICING

For servicing of Acorn products - use any authorised Acorn dealer or Acorn's appointed service agent:

R.C.S.
Gresham House
Twickenham Road
Feltham
Middlesex
[Tel: 01-898 4761]

TECHNICAL AND OTHER ENQUIRIES (ACORN)

Technical and other enquiries including complaints.

Acorn Computers
Fulbourn Road
Cherry Hinton
Cambridge CB1 4JN
[Tel 0223 210111]

Note. Acorn have installed many new lines, and you should now have no trouble getting through. If there is trouble, use their old number 0223 245200 which is now under much less pressure.

BBC REFERRAL SERVICE

For general information as part of the Computer Literacy Project send a large SAE to:

BBC Referral Service
PO Box 7
London W3 6XJ

UNRESOLVED COMPLAINTS

Also unresolved complaints should be addressed to:

The Head of Merchandising
BBC Enterprises Ltd
The Langham
Portland Place
London W1A 1AA

TECHNICAL ENQUIRIES (BBC)

Any technical enquiries that you wish to address to the BBC rather than Acorn should be sent to:

Colin Malone
BBC Enterprises Ltd
Merchandising
The Langham
Portland Place
London W1A 1AA

BEEBUG SUBSCRIPTIONS AND SOFTWARE

Note our new address for both software AND subscriptions is now:

BEEBUG
PO BOX 109
Baker Street
High Wycombe
Bucks HP11 2TD

TWO EPROM PROGRAMMERS REVIEWED

by Rob Pickering

Rob Pickering chooses two Eprom programmers to review, and compares their facilities and features.

During the course of 1983 we will increasingly be seeing software available in EPROMs and ROMs, as the basic production costs fall and the general awareness increases. An EPROM programmer is an essential device for anyone wishing to put their own software into EPROM, so we have taken a look at the current state of the market. Two companies have kindly supplied machines for review. No background information on EPROMs or ROMs is given, though you may like to refer to page 16 of the March 1983 Hobby Electronics magazine which explains the technology quite clearly.

The market for EPROM programmers can be identified mainly as: small businesses, education, an increasing number of hobbyists, and software Pirates! The ease with which EPROMs can be copied illegally is rather disconcerting, since it will potentially lead to less professional software being produced.

Vacant sockets inside the machine, situated adjacent to the Operating System, are generally referred to as the paged ROM (or sideways ROM) sockets. However it is possible to put a variety of things into EPROMs in these sockets. Basic programs, data files etc. can all be stored, then used by typing *ROM. After an *ROM command subsequent 'read' operations such as LOAD"" will read from the language ROM sockets, instead of the cassette or discs. (The *ROM command is only available on the series 1 OS).

REVIEW MACHINE from ATPL

SUPPLIED BY: ATPL, Station Road, Clowne, Chesterfield. S43 4AB.
PRICE: £120 +VAT, inc.P&P
INCLUDES: Software cassette, all connectors, manual.
EPROM SIZES USABLE: 4,8,16k bytes (note: kbytes, not kbits)

The ATPL unit is very well presented indeed. With a well written, well printed A5 manual (27 pages); a cassette wallet containing one cassette with all required software; and the device itself ready to plug in. It connects to both the 1MHz Bus via a 34-way speedblock connector on ribbon cable, and to the power output socket on the Beeb (make sure your's has one fitted).

The software used to control the programmer is on the cassette and must first be loaded and executed. The programming related functions are: Blow, Read, Verify, Clear. 'Blow' programs an EPROM from RAM contents; Read reads an EPROM's contents into RAM; Verify checks that the EPROM has programmed correctly; Clear checks that the EPROM is blank. The Blow command automatically carries out a 'clear' check before programming, just in case you've accidentally inserted an EPROM which is already programmed. I would have overwritten several EPROMs were it not for this safety check.

I was very impressed with the software when I first received it, and remain impressed after using it a great deal. The software is written entirely in machine code and a full assembly listing (the source code) is provided both as a 5 page listing in the manual, and also on cassette. The fact that it is on cassette means that it can be loaded and re-assembled at a new location. I had to re-assemble it at &1900 in order to use it from discs, for which the instructions are given in the

manual. It could of course be located at the top end of memory. The software rates highly on ease of use, with commands to do just what you need. In addition to the four main commands, operating system commands starting with an asterisk are recognised and carried out. Thus, data to be programmed into the EPROM can be loaded form disc or cassette with *LOAD giving the RAM location at which it is to be loaded.

All that has been mentioned so far is the programming capability, but the ATPL device has another side to it as well. Any Basic or machine code program may be programmed into one or two EPROMs (a maximum length between 4-32k long) and then placed into either or both of the two special sockets underneath the unit. This program may then be loaded and automatically executed merely by pressing BREAK. It only takes a second or two for the program to load and execute, so speed is comparable to discs. Four switches are provided on the top of the enamelled metal case which are set accordingly for OS 0.1 or series 1, Basic program, machine code program, or no program at all. Nothing else needs be done. With the switches set and the device plugged in, switching on the power to the BBC micro will result in the program being loaded and executed in an instant. In order to achieve this 'auto run' facility, it is necessary to take a machine code routine supplied on the cassette and program it into an EPROM. This caused me some initial delay because I didn't know I needed the blank EPROM. However, ATPL do state this on their order form and in their data sheets and can supply a suitable device. It is worth noting this, though if you order straight from a magazine and without seeing an order form, remember that you need a 2732 EPROM.

DRAWBACKS: Only one program may be in the device for auto-run at any one time. The use of the power supply socket, used by many types of disc drive, cannot be used for both programmer and discs at the same time. ATPL rightly point out that not many people will want to use both. The connecting cables on the unit are very short and make it difficult to position conveniently for use. It can only sit very close in front or to the right of the machine. The most obvious disadvantage is the cost, at about £138 inclusive, it is beyond the reach of most 'hobbyists' who are merely interested.

REVIEW MACHINE from MED

SUPPLIED BY: MED, 640 Melton Rd., Thurmaston, Leicester, LE4 8BB.
PRICE: £79.90 fully inclusive.
INCLUDES: Software cassette, all connectors except mains plug, instructions.
EPROM SIZES USABLE: 2,4k bytes only.

The MED unit itself is well made, but falls short of the ATPL unit in many ways, though it must be pointed out right at the start that it is much cheaper too. The documentation is poorly presented as A4 copies of dot-matrix printer output. However, the content is good enough to help an experienced user to cope with the software.

The programming software is not at all as easy to use, requiring a lot more thought and understanding. However, its relative complexity includes the possibility of reading or writing any specified part of the EPROM, whereas the ATPL unit deals only in terms of the entire EPROM at once.

Because there is only a capability to program 2k and 4k EPROMs, its use for programming EPROMs for the BBC micro is somewhat limited, though for other micro-electronics applications such as in older computers like UK101, PET, etc. it is perfectly adequate. The future of programming EPROMs for the BBC micro is likely to be with 8k and even 16k devices. However, MED have informed me that they are going to produce an add-on to cope with 8k EPROMs and possibly 16k eproms.

Since there is no auto-run facility at all on this unit, it is not possible to immediately use an EPROM which you have programmed for the BBC micro. Although MED have successfully placed a 4k EPROM in one of the internal ROM sockets, it was necessary to use an extra machine code routine in order to transfer the EPROMs contents into RAM. All in all a rather untidy and experimental method at present.

CONCLUSIONS: The two machines are not really aimed at quite the same people as far as I can tell. The ATPL unit is useful to those wanting to program EPROMs quickly and easily, and/or to use them for the BBC micro. The MED unit seems more of an investigatory and development device, but for producing EPROMs for use with other machines... for which it is very good. Finally, the difference in size of EPROM which can be programmed is a major difference between the two, but bear in mind the following: although costs are falling rapidly, a 16k eprom still retails at about still retails at 4 to 5 times the price of an 8k EPROM.

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

FUNCTIONS AND PROCEDURES - Bjorn Grand

It is possible to use functions and procedures from 'immediate mode', provided that the function or procedure definition is in the current program in memory e.g.

```
10 DEFFNFAK(N%)        for example, typing
20 IF N%=0 THEN=1      PRINT FNFAK(3) gives
30 =N%*FNFAK(N%-1)     an answer of 6.
```

This allows procedures and functions to be written and tested before the rest of the program is written. Thus you can use BBC Basic a little like Forth, writing the small routines first, followed by their linking together. This should be useful to the structured programming freaks.

OSFIND - Bjorn Grand, Matthew Rapier

According to both the BBC User guide, and the Disc System User Guide, OSFIND returns the file handle in Y. In fact it returns the file handle in A!

Another bug in the manual is the swapping of much information in the technical section of the Disc Sytem User Guide. The various addresses stored in the catalogue (sector 01) are in fact stored low order bits followed by middle order bits.

PROGRAM TRANSFER BETWEEN BBC AND 380Z (New OS) - P.J.Andrews

With the new operating system it is extremely easy to tranfer data between these two computers down the serial port. To do this, first set both input and output baud rates on both machines to the same value, *FX7 and *FX8 on the BBC.

To send data to the 380Z, treat the serial link-up exactly as you would a printer from the BBC. CP/M requires a Ctrl-Z character as the terminator of the file. Use VDU 1,26,3 to send this. The file can be read by PIP or TXED at the 380Z end.

To transfer programs back again, use *FX2,1 on the BBC to accept data from the RS423 as keyboard input. Data can be sent by PIP, or TXED (or LLIST in 380Z Basic). Entry can be terminated by pressing BREAK on the BBC, followed by OLD. It should be possible for the 380Z to send *FX2,0 or something similar to end the BBC's reception. [This hint will only make sense to users of CP/M and/or the Research Machines 380Z computer.]

For O.S. 1.2 Only

THE NEW 1.2 OS COMMANDS *CODE AND *LINE

by Colin Browell

Last month we gave notes on two undocumented commands *CODE and *LINE available on OS 1.2. Here we explain how to use them in a little more detail.

If you have ever tried to disassemble the 1.2 Operating System ROM, you will probably have come across the OS command table. This is located at &DF14 onwards and contains a list of 'star commands' which are recognised by the operating system itself, such as *CAT and *H.
You may also have noticed that there are two undocumented commands present, *LINE and *CODE: but what do they do? If you type them in directly they always produce a 'Bad Command' message, as if the operating system does not recognise them. The reason for this is that *LINE and *CODE require the setting up of an 'indirection vector' before they can be used.

An indirection vector consists of two consecutive bytes of RAM which contain the entry address of a machine code routine. For instance, when a character is to be printed to the screen, the OS routine OSWRCH must be called: the address of this routine in the 1.2 ROM is &E0A4. But rather than jump DIRECTLY to this address, an INDIRECT jump is performed to the address which is stored in the OSWRCH indirection vector.

One of the advantages of this method of accessing routines is that by changing the contents of the vector, the user can intercept any call to that routine and force the computer to execute his own code (see User Guide P.457). A table can be found on P.425 of the User Guide which lists the major operating system routines and their respective vectors.

One of the vectors, USERV, is free for the programmer's use, and is used by *LINE and *CODE. This explains why these commands always seem to invoke the 'Bad Command' message - rather than allowing the vector to 'flap around', pointing at no particular address, it is 'tied down' to &E310 whenever a Reset occurs. As this is the address where the 'Bad Command' message is stored, *LINE and *CODE will always invoke this error until USERV is set up.

The two commands can be described thus:

*CODE X,Y (Where X and Y are integers between 0 and 255)

On executing this command, the specified values X and Y are loaded into the X and Y registers respectively, and the accumulator is loaded with zero. An indirect jump is then performed to the address pointed to by USERV: the user's routine should be located at this address. Thus:
?USERV=&EA:?(USERV+1)=&40:*CODE 214,56
would be equivalent to:
X%=214 : Y%=56 : A%=0 : CALL &40EA

*LINE s (Where s is a string of characters - NOT enclosed by quotation marks)

On executing this command, the address of the first character of the string is loaded into the X and Y registers (X:lo byte, Y:hi byte), and the accumulator is loaded with 1.

Thus *CODE and *LINE call a user supplied routine whose address has previously been deposited in USERV. *CODE allows the user to specify the contents of the X and Y registers on entry, and *LINE allows the routine to access a data string.

A couple of examples may help to clarify their usage. Listing 1 shows an application of *LINE to insert characters into the keyboard buffer: line 360 is particularly important as it demonstrates that Basic commands such as LIST and NEW which could not normally be used in a program, can be executed when the program ends.

Listing 2 demonstrates that a routine can handle both *LINE and *CODE commands. This is because *CODE loads the accumulator with Ø, while *LINE loads it with 1. Thus lines 16Ø and 17Ø check to see if the accumulator contains zero: if it does then the routine continues with the *CODE handling part, otherwise it branches to the *LINE handling part.

On encountering a *CODE instruction, the routine positions the cursor at X,Y (i.e. the two *CODE parameters); and on a *LINE instuction, the string 's' is written to the screen. Thus *CODE performs an X,Y function, and *LINE performs a simple string printing function.

Try adding teletext contol codes to the strings, and then RUN the program again. The Basic interpreter will consider the codes to be keyword tokens, and therefore any codes you insert will list as Basic keywords: for instance 'Alpha Red' (=&81) will list as 'DIV', but the single byte &81 is still present in memory, so when the program is run again the 'Alpha Red' character is still printed.

```
 10 REM ....Listing 1....
 20 REM
 30 REM *LINE DEMO PROGRAM
 40 REM by C Browell v1A
 50
 60 DIM assem_address% 35
 70 userv%=&200
 80 osbyte%=&FFF4
 90
100 ?userv%=assem_address% MOD 256
110 userv%?1=assem_address% DIV 256
120
130 FOR PASS=0 TO 2 STEP 2
140 P%=assem_address%
150 [OPT PASS
160 STX&70
170 STY&71
180 LDX#0
190 STX&72
200 .loop
210 LDY&72
220 LDA(&70),Y
230 INY
240 STY&72
250 TAY
260 LDA#&8A
270 CPY#13
280 BEQlastbyte
290 JSR osbyte%
300 JMPloop
310 .lastbyte
320 JMP osbyte%
330 ]NEXT
340
350 *LINE MODE6:VDU19,0,4;0;14
360 *LINE LIST
370
380 END
```

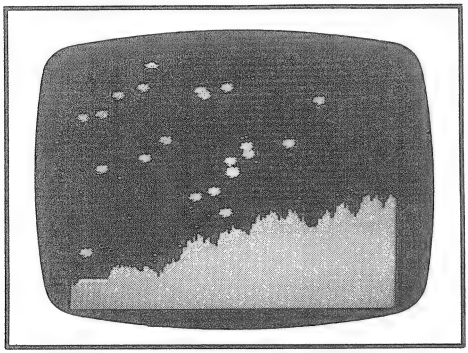
```
 10 REM ....Listing 2....
 20 REM
 30 REM *CODE/*LINE DEMO PROGRAM
 40 REM by C Browell v1A
 50
 60 DIM assem_address% 100
 70 userv%=&200
 80 oswrch%=&FFEE
 90
100 ?userv%=assem_address% MOD 256
110 userv%?1=assem_address% DIV 256
120
130 FOR PASS=0 TO 2 STEP 2
140 P%=assem_address%
150 [OPT PASS
160 CMP#0
170 BNE star_line
180 LDA#31
190 JSR oswrch%
200 TXA
210 JSR oswrch%
220 TYA
230 JSR oswrch%
240 .end_command
250 RTS
260 .star_line
270 STX&70
280 STY&71
290 LDY#0
300 .loop
310 LDA(&70),Y
320 CMP#13
330 BEQ end_command
340 INY
350 JSR oswrch%
360 JMP loop
370 ]NEXT
380 MODE7
390
400 *CODE 9,8
410 *LINE "COMMAND LINE
420 *CODE 14,11
430 *LINE INTERPRETER"
440 *CODE 19,14
450 *LINE RULES,      OK!!
460 *CODE 0,23
470
480 END
```

Tested on O.S. 0·1 and 1·2 and on Basics I and II

MARS LANDER (16k)

by Alan Webster

Mars Lander is a version of the popular moon lander game in which the player attempts to navigate his space craft through a mass of meteors to land it softly on the landing pad. You control the horizontal and vertical speed of the craft. The game runs in mode 5 and makes good uses of both colour and sound.

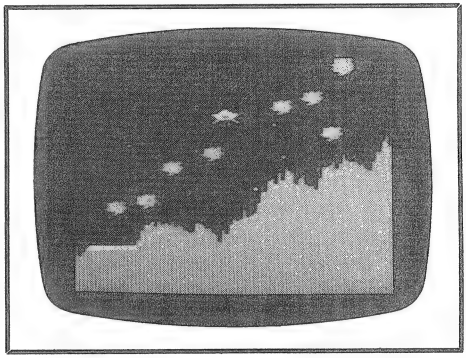


The player can choose the level of difficulty for the game - the more difficult the game the more sensitive the controls. The roughness of the terrain can also be chosen, ranging from quite smooth to very rough. You have 15 seconds to land the craft before your fuel runs out.

The controls are given in the program, and for those of you who find the game too easy (in excess of 800 points) you might like to change the gravity factor by altering line 160 to:

```
160 N%=N%-(G%/1.95-(1/(Y%
    /10))):IF N%>80 N%=80
```

There is also a 32k version of Mars Lander on the magazine cassette. This runs in mode 2 and uses more colours (and even has flashing stars!)
The alteration to make this version harder is: 370 N%=N%-(G%/1.95-(1/(Y%/10)))



The picture that accompanies this program is of the 32k version, but of course, we cannot show the stars flashing!

```
  10 REM Mars Lander Version 1A
  20 ON ERROR GOTO 560
  30 PROCchars:PROCarrays:MODE4
  40 PRINTTAB(14,2)"Mars Lander";TAB(1
2,3)"by Alan Webster"
  50 PRINTTAB(5,10)"Difficulty (1=Easy
,8=Hard)":G%=GET:IFG%<49 OR G%>56 GOTO
50
  60 G%=G%-47:dead=FALSE:SC%=0
  70 PRINTTAB(5,12)"Terrain (1=Smooth,
8=Rough)":L%=GET:IF L%<49 OR L%>56 GOTO
 70
  80 L%=L%-48:L%=L%*6:PRINTTAB(13,19)"
Z - Left";TAB(13,20)"X - Right";TAB(8,2
1)"RETURN - Thrust";TAB(9,23)"Press any
 key":i=GET
  90 MODE5:A2%=100:VDU5:REPEAT:CLS:PRO
Cmeteors:X%=RND(900):Y%=1000:M%=0:N%=1:
landed=FALSE:PROCscene
 100 TIME=0:REPEAT:OX%=X%:OY%=Y%
 110 IFINKEY-98 IF X%>32 M%=M%-G%
 120 IFINKEY-67 IF X%<1210 M%=M%+G%
 130 X%=X%+(M%/2):IFX%<32 X%=32
 140 IFX%>1210 X%=1210
 150 M%=M%+(SGN(M%)/10):IFINKEY-74 N%=
N%+G%
 160 N%=N%-(G%/10):IFN%>80 N%=80
 170 Y%=Y%-8+N%:IFOX%=X% IF OY%=Y% GOT
O 190
 180 MOVEOX%,OY%:GCOL0,0:VDU224:MOVEX%
,Y%:GCOL0,3:VDU224
 190 R%=POINT(X%+55,Y%-31):R1%=POINT(X
%,Y%-31):SOUND0,-10,4,1:IFR%=1 dead=TRUE
 200 IFR1%=1 dead=TRUE
 210 IFR%=2 IFR1%=2 landed=TRUE
 220 IF TIME>1500 dead=TRUE
 230 UNTIL landed OR dead:IFN%<1 dead=
TRUE
```

```
  240 G%=G%+1:IF dead GOTO260
  250 SC%=SC%+(1500-TIME):FORSO%=1TO10
:SOUND1,-10,200,1:SOUND1,-10,100,1:NEXT
  260 L%=L%+4:UNTIL dead:SC%=SC%-100:PR
OCexpl:FORY=1TO3000:NEXT:*FX15
  270 MODE7:PRINTTAB(5,10)" Score was :
";SC%:FORY=1TO5000:NEXT:RUN
  280 DEFPROCscene
  290 P%=RND(120)*8:FORA1%=0TO1280STEP8
:IFA1%=P% S%=A2%
  300 IFA1%>P% IFA1%<(P%+120) PROCpad:G
OTO330
  310 GCOL0,1:MOVEA1%,0:DRAWA1%,A2%:A2%
=A2%+(RND(L%)-(L%/2)):IFA2%<20A2%=20
  320 IFA2%>400A2%=400
  330 NEXT
  340 ENDPROC
  350 DEFPROCpad
  360 GCOL0,2:PLOT69,A1%,S%:PLOT69,A1%,
S%+4:GCOL0,1:MOVEA1%,0:DRAWA1%,S%-4:GCO
L0,0:MOVEA1%,S%+8:DRAWA1%,S%+72
  370 ENDPROC
  380 DEFPROCchars
  390 VDU23,224,0,24,60,255,102,60,66,1
29,23,225,20,126,124,126,254,124,28,48
  400 ENDPROC
  410 DEFPROCarrays
  420 ENVELOPE 1,3,0,0,0,0,255,0,127,0,
0,-127,126,126
  430 DIMZ%(16),Z2%(16),Q%(16),Q1%(16)
  440 FOR Z1%=1TO16:Q%(Z1%)=RND(12):IFQ
%(Z1%)>6 Q%(Z1%)=Q%(Z1%)-13
  450 Q1%(Z1%)=RND(12):IFQ1%(Z1%)>6 Q1%
(Z1%)=Q1%(Z1%)-13
  460 NEXT
  470 ENDPROC
  480 DEFPROCexpl
  490 GCOL0,0:MOVEX%,Y%:VDU224:FOR Z6%=
1TO16:Z%(Z6%)=X%:Z2%(Z6%)=Y%:NEXT
  500 FORP%=1TO14:FORJ%=1TO16:GCOL0,0:P
LOT69,Z%(J%),Z2%(J%):Z%(J%)=Z%(J%)+(Q%(
J%)*4)
  510 GCOL0,1:Z2%(J%)=Z2%(J%)+(Q1%(J%)*
2):GCOL0,7:PLOT69,Z%(J%),Z2%(J%):NEXT:S
OUND 0,1,6,5:NEXTP%
  520 ENDPROC
  530 DEFPROCmeteors
  540 GCOL0,1:FORW%=1TO(L%/2.5)+RND(5):
MOVE RND(1000),RND(600)+200:VDU225:NEXT
  550 ENDPROC
  560 ONERROROFF
  570 MODE7:IF ERR=17 END
  580 REPORT:PRINT " at line ";ERL
  590 END
```

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

HEAD DEMAGNETISING - Daniel Cooke

While a tape recorder is being used, the heads of the recorder slowly become magnetised. When the magnetic field builds up it has the effect of wiping all your tapes for you! A head demagnetiser can be used to remove this destructive field. Proprietary head demagnetisers cost a few pounds, but are well worth the expense. Just consider the value of all your programs. It is of course a good idea to keep back-up copies of your tape programs, most people only bother taking back up copies of discs. It is equally relevant with tapes.

OPENIN OPENOUT OPENUP - Rik Bean, Matthew Rapier

The Basic filing commands are rather confusing as Acorn have changed the meanings of the OPEN words in Basic II. The commands are very much simpler in Basic II. Obviously the random access capability is only usable from the DFS.

In Basic I, OPENIN opens a file for input and output. OPENOUT creates a file for output only. This is rather strange as the OSFIND call supports a purely input state as well.

In Basic II, OPENIN is purely for input, OPENUP for input and output, and OPENOUT to create a file for output only.

Thus in Basic II OPENUP is the same as the OPENIN in Basic I. You should note that Acorn have given the token for OPENIN in Basic I the word OPENUP in Basic II. Thus a Basic I program can be loaded on a Basic II machine, and will work perfectly, with OPENIN replaced by OPENUP. But if the program is typed in using Basic II, and you will later want to use it on a Basic I machine you MUST not use OPENIN, but use OPENUP instead.

Tested on O.S. 0·1 and 1·2 and on Basics I and II

SPACE LORDS (32k)

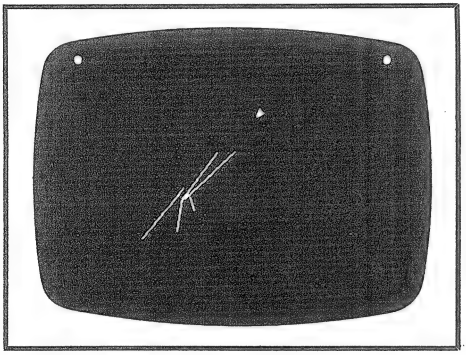
by D J Hoskins

FOR TWO PLAYERS

SPACE LORDS is great fun to play, and is one of the few two player games around. It is loosely based on the game of "asteroids", except in Space Lords each player moves his spaceship around the screen trying to blast his opponent's ship. The game runs in mode 1 and uses graphics and sound very well.

Instructions for play are included in the program and are given when the game is run. Briefly though, each player uses a set of four keys on each side of the keyboard to control his ship. Controls are similar to those used in most other asteroids type games.

If you are running the program from disc, you must set PAGE to &1200 before CHAINing it. For example: PAGE=&1200:CHAIN"SPLORDS"



```
  10 REM Space-Lords
  20 REM by D.J.Hoskins
  30 REM Version 1A
  40 *TV0,1
  50 *FX4,4
  60 ONERRORGOTO1460
  70 PROCSET:F%=0
  80 MODE7:VDU23;8202;0;0;0;0:PROCINST
  90 MODE1:VDU23;8202;0;0;0;0;19,1,6;0;
 100 GCOL0,3:FORF=1TO180:PLOT69,RND(12
79),RND(1023):NEXT
 110 COLOUR1:PRINTTAB(0,1);S1%:COLOUR2
:PRINTTAB(39,1);S2%
 120 VDU5
 130 IFF%<>2A%=RND(600):B%=RND(750)+10
0:GC=1:PROCMAT(A%,B%):L=S
 140 IFF%<>1X%=RND(560)+680:Y%=RND(750
)+100:GC=2:PROCMAT(X%,Y%):R=S
 150 LL=L:RR=R:A=0:B=0:X=0:Y=0
 160 IFS1%=0ANDS2%=0RESTORE1230:FORF=1
TO17:READF%,G%:SOUND&11,4,F%,16:SOUND&1
2,4,F%-48,16:SOUND&13,4,F%,16:FORG=1TO
(G%*90):NEXTG,F
 170 IFINKEY(-98)=-1L=L-1:IFL=0L=8
 180 IFINKEY(-67)=-1L=L+1:IFL=9L=1
 190 IFINKEY(-97)<>-1GOTO240
 200 IFL=1ORL=2ORL=8B=B+4 ELSEIFL<>3AN
D L<>7B=B-4
 210 IFL=2ORL=3ORL=4A=A+4 ELSEIFL<>1AN
D L<>5A=A-4
 220 A=A MOD58:B=B MOD58
 230 SOUND&10,-13,7,5:SOUND&11,0,190,5
 240 GCOL3,1:MOVEA%,B%:IFLL<>L VDU223+
LL ELSE VDU223+L
 250 A%=A%+A:B%=B%+B:MOVEA%,B%:VDU223+ L
 260 IFINKEY(-90)=-1R=R-1:IFR=0R=8
 270 IFINKEY(-106)=-1R=R+1:IFR=9R=1
 280 IFINKEY(-42)<>-1GOTO330
 290 SOUND&10,-13,7,5:SOUND&11,0,150,5
 300 IFR=1ORR=2ORR=8Y=Y+4 ELSEIFR<>3AN
D R<>7Y=Y-4
 310 IFR=2ORR=3ORR=4X=X+4ELSEIFR<>1AND
 R<>5X=X-4
 320 X=X MOD58:Y=Y MOD58
 330 GCOL3,2:MOVEX%,Y%:IFRR<>R VDU223+
RR ELSEVDU223+R
 340 X%=X%+X:Y%=Y%+Y:MOVEX%,Y%:VDU223+
R:GCOL3,3
 350 IFINKEY(-2)=-1PROCLINT
 360 IFINKEY(-74)=-1PROCRINT
 370 IFQ%=0GOTO410
 380 MOVEN%,M%:VDU255:N%=N%+N:M%=M%+M:
MOVEN%,M%:VDU255
 390 IFN%>1279ORN%<-20ORM%<-20ORM%>105
4Q%=0
 400 IFN%>X%-36ANDN%<X%+24ANDM%>Y%-34A
NDM%<Y%+26F%=2:GOTO820
 410 IFW%=0GOTO450
 420 MOVEJ%,K%:VDU255:J%=J%+J:K%=K%+K:
MOVEJ%,K%:VDU255
 430 IFJ%>1279ORJ%<-20ORK%<-20ORK%>105
4W%=0
 440 IFJ%>A%-36ANDJ%<A%+24ANDK%>B%-34A
NDK%<B%+26F%=1:GOTO820
 450 IFB>.2B=B-.5ELSEIFB<0B=B+.5
 460 IFA>.2A=A-.5ELSEIFA<0A=A+.5
 470 IFX>.2X=X-.5ELSEIFX<0X=X+.5
 480 IFY>.2Y=Y-.5ELSEIFY<0Y=Y+.5
```

```
  490 IFX%<1290ANDX%>-32ANDY%>-24ANDY%<
1060GOTO520
  500 IFX%>1290X%=-30 ELSEIFX%<-32X%=12
84
  510 IFY%>1060Y%=-24 ELSEIFY%<-24Y%=10
60
  520 IFA%<1290ANDA%>-32ANDB%>-24ANDB%<
1060GOTO550
  530 IFA%>1290A%=-30 ELSEIFA%<-32A%=12
84
  540 IFB%>1060B%=-24 ELSEIFB%<-24B%=10
60
  550 LL=L:RR=R
  560 GOTO170
  565
  570 DEFPROCLINT
  580 IFQ%<>0MOVEN%,M%:VDU255
  590 N=0:M=0:Q%=1:N%=A%:M%=B%:SOUND&13
,1,150,2:MOVEN%,M%:VDU255
  600 IFL=1ORL=2ORL=8M=48 ELSEIFL<>3AND
 L<>7M=-48
  610 IFL=2ORL=3ORL=4N=48 ELSEIFL<>1AND
 L<>5N=-48
  620 ENDPROC
  625
  630 DEFPROCRINT
  640 IFW%<>0MOVEJ%,K%:VDU255
  650 J=0:K=0:W%=1:J%=X%:K%=Y%:SOUND&12
,1,180,2:MOVEJ%,K%:VDU255
  660 IFR=1ORR=2ORR=8K=48 ELSEIFR<>3AND
 R<>7K=-48
  670 IFR=2ORR=3ORR=4J=48 ELSEIFR<>1AND
 R<>5J=-48
  680 ENDPROC
  685
  690 DEFPROCMAT(I%,E%):GCOL3,GC
  700 VDU29,I%+16;E%-16;
  710 SOUND&11,0,0,0:SOUND&10,-15,7,255
  720 FORF=0TO9:U%(0,F)=(RND(30)-15)*25
0:U%(1,F)=(RND(30)-15)*250:Q=U%(0,F):W=
U%(1,F):PLOT69,Q,W:PLOT69,Q+4,W:PLOT69,
Q,W+4:PLOT69,Q+4,W+4:NEXT
  730 FORG=1TO3:FORF=0TO9:Q=U%(0,F):W=U
%(1,F)
  740 SOUND&11,0,20+(G*10+F)*4,2
  750 PLOT69,Q,W:PLOT69,Q+4,W:PLOT69,Q,
W+4:PLOT69,Q+4,W+4
  760 Q=Q*.2:W=W*.2
  770 PLOT69,Q,W:PLOT69,Q+4,W:PLOT69,Q,
W+4:PLOT69,Q+4,W+4:U%(0,F)=Q:U%(1,F)=W
  780 NEXT:NEXT:GCOL0,0
  790 FORF=0TO9:Q=U%(0,F):W=U%(1,F):PLO
T69,Q,W:PLOT69,Q+4,W:PLOT69,Q,W+4:PLOT6
9,Q+4,W+4:SOUND&11,0,F*20,5:NEXT
  800 GCOL3,GC:VDU29,0;0;:MOVEI%,E%:S=R
ND(8):VDU223+S
  810 SOUND&11,0,0,0:SOUND&10,0,0,0:END
PROC
  820 IFF%=1I%=A%:E%=B%:P%=L
  830 IFF%=2I%=X%:E%=Y%:P%=R
  840 GCOL3,3:IFF%=2ANDQ%<>0MOVEN%,M%:V
DU255:Q%=0ELSEIFW%<>0ANDF%=1MOVEJ%,K%:V
DU255:W%=0
  850 GCOL3,F%:MOVEI%,E%:VDU223+P%
  860 VDU29,I%+16;E%-16;
  870 SOUND&11,2,200,255:SOUND&10,-15,7
,255
  880 FORF=0TO9STEP2:U%(0,F)=(RND(100)-
50)*2:U%(1,F)=(RND(100)-50)*2:U%(0,F+1)
=RND(40)-20:U%(1,F+1)=RND(40)-20:MOVE U
%(0,F),U%(1,F):DRAW U%(0,F+1),U%(1,F+1)
:NEXT
  890 FORG=1.5TO1.2STEP-.05:FORF=0TO9ST
EP2
  900 MOVE U%(0,F),U%(1,F):DRAW U%(0,F+
1),U%(1,F+1)
  910 U%(0,F)=U%(0,F)*G:U%(1,F)=U%(1,F)
*G:U%(0,F+1)=U%(0,F+1)*G:U%(1,F+1)=U%(1
,F+1)*G
  920 MOVE U%(0,F),U%(1,F):DRAW U%(0,F+
1),U%(1,F+1):NEXTF,G
  930 SOUND&11,0,0,0:SOUND&10,0,0,0:FOR
F=0TO9STEP2:MOVE U%(0,F),U%(1,F):DRAW U
%(0,F+1),U%(1,F+1):NEXTF
  940 *FX15,0
  950 IFF%=2S1%=S1%+1ELSES2%=S2%+1
  960 VDU4:COLOUR1:PRINTTAB(0,1);S1%:CO
LOUR2:PRINTTAB(39,1);S2%
  970 FORF=1TO15:G=200+RND(15):SOUND&11
,3,G,2:SOUND&12,3,G-48,2:FORH=1TORND(12
0):NEXTH,F
  980 VDU29,0;0;
  990 FORF=1TO3000:NEXT
 1000 IFS1%<3ANDS2%<3GOTO110
 1010 RESTORE1240:FORF=1TO33:READH,G%:S
OUND&11,4,H-48,16:SOUND&12,4,H-48,16:FO
RG=1TO (G%*70):NEXTG,F
 1020 FORF=1TO3000:NEXT
 1030 PRINTTAB(0,0);:FORF=0TO30:SOUND&1
2,-15,RND(255),1:VDU11:SOUND&11,-15,RND
(255),1:SOUND&13,-15,RND(255),1:FORG=1T
025:NEXTG,F
 1040 MODE7:VDU23;8202;0;0;0;0
 1050 A$="       THE ":IFF%=1A$=A$+"RIGH
T "ELSEIFF%=2A$=A$+"LEFT "
 1060 A$=A$+"HAND PLAYER WON.       "
 1070 FORF=1TOLENA$:PRINTTAB(0,9);CHR$&
86;RIGHT$(A$,F):SOUND&11,-15,ASC(RIGHT$
(A$,F)),1:FORG=1TO40:NEXTG,F
 1080 *FX15,0
 1090 PRINTTAB(0,20);CHR$157;CHR$&84;"
       PRESS SPACE TO START"
 1100 IFINKEY(0)<>32GOTO1100
 1110 S1%=0:S2%=0:CLG:F%=0:Q%=0:W%=0:GO
TO90
 1120 END
 1125
 1130 DEFPROCSET
 1140 VDU23,224,24,24,36,36,66,90,165,1
95,23,225,3,13,50,194,52,20,8,8,23,226,
192,176,76,35,35,76,176,192
```

```
 1150 VDU23,227,8,8,20,52,194,50,13,3,2
3,228,195,165,90,66,36,36,24,24,23,229,
16,16,40,44,67,76,176,192
 1160 VDU23,230,3,13,50,196,196,50,13,3
,23,231,192,176,76,67,44,40,16,16,23,25
5,0,0,0,24,24,0,0,0
 1170 DIMU%(1,10):S1%=0:S2%=0:Q%=0:W%=0
 1180 ENVELOPE1,1,-10,-5,-2,2,10,300,0,
0,0,-3,126,0
 1190 ENVELOPE2,2,-1,-1,-1,200,200,200,
0,0,0,-126,0,0
 1200 ENVELOPE3,1,-7,-7,-7,200,200,200,
0,0,0,-5,126,0
 1210 ENVELOPE4,1,0,0,0,200,200,200,0,-
1,0,-126,126,0
 1220 ENDPROC
 1225
 1230 DATA101,5,121,2,129,2,137,4,137,4
,129,4,141,4,137,4,101,4,93,4,137,2,121
,2,129,4,93,4,89,4,117,4,121,16
 1240 DATA149,6,149,4,137,2,141,2,149,4
,169,4,165,4,149,4,149,4,149,4,137,2,14
1,2,149,4,169,4,165,4,149,4,149,4,169,4
,185,2,177,2,169,4,149,4,157,4,157,2,14
9,2,157,4,129,4,129,4,137,4,141,4,149,4
,137,4,121,8
 1245
 1250 DEFPROCINST
 1260 FORF=1TO2:PRINTTAB(5,F);CHR$141;C
HR$129;CHR$136;">>]]]]SPACE-LORDS[[[[<<
":NEXT
 1270 PRINTTAB(0,3);CHR$&84;"V. 1.2";CH
R$&82;" ";CHR$&84;"David J. Hoskins."
 1280 PRINTTAB(0,5);CHR$&86;"This is a
TWO player game of skill and ";CHR$&86;
"strategy.";TAB(0);CHR$&83;"The object
 of the game is to blow your ";CHR$&83;"
opponent up three times to win."
 1290 PRINTCHR$&82;"The two space ships
 drift in space"SPC(5);CHR$&82;"thruste
d around by the keys:";TAB(0);CHR$&86;"
""TAB""-LEFT HAND PLAYER(CYAN)";TAB(0);
CHR$&83;"""CURSOR DOWN""-RIGHT HAND PLAY
ER(YELLOW)"
 1300 PRINTTAB(0,13);CHR$&85;"When thru
sting you go in the direction ";CHR$&85
;"you are pointing."
 1310 PRINTCHR$&81;"To rotate the space
 ships use the keys:";"          FOR THE
LEFT HAND PLAYER.......";TAB(0);CHR$&86
;"""Z""-ROTATE LEFT  ""X""-ROTATE RIGHT
"
 1320 PRINT"          FOR THE RIGHT HAND
PLAYER......"
 1330 PRINTCHR$&83;"""DELETE""-ROTATE L
EFT  ""COPY""-ROTATE    ";CHR$&83;"RIGH
T"
 1340 PRINTCHR$&85"NB- IF YOU GO OFF TH
E SCREEN YOU WILL  ";CHR$&85;"APPEAR ON
 THE OTHER SIDE."
 1350 PRINT"          PRESS SPACE TO CON
TINUE"
 1360 IFINKEY(0)<>32GOTO1360 ELSECLS
 1370 FORF=1TO2:PRINTTAB(5,F);CHR$141;C
HR$130;CHR$136;">>]]]]SPACE-LORDS[[[[<<
":NEXT
 1380 PRINTTAB(2,4);"To fire use the ke
ys:";TAB(4);CHR$&86;"""CTRL""-LEFT HAND
 PLAYER.";TAB(4);CHR$&83;"""RETURN""-RI
GHT HAND PLAYER."
 1390 PRINT:PRINT"* IF YOU FIRE WHILE Y
OUR LASER IS STILL IN THE AIR YOUR FIRS
T ONE WILL DISAPPEARAND A NEW ONE WILL
BE FIRED."
 1400 PRINT:PRINTCHR$&81;"When a player
 is ZAPPED he/she will    ";CHR$&81;"ma
terialise randomly on the screen."
 1410 PRINTCHR$&86;"If you go too fast
your engines will    ";CHR$&86;"ABORT,st
opping you instantly. But not  ";CHR$&8
6;"for GOOD -You can quickly regain pow
er."
 1420 PRINTTAB(0,17);CHR$&82;"The space
 ships cannot collide, and you";CHR$&82
;"cannot be hit by your own laser."
 1440 PRINTTAB(0,23)"          PRESS SPA
CE TO BEGIN"
 1450 REPEAT UNTIL GET=32:ENDPROC
 1460 ONERROROFF
 1470 *FX4
 1480 MODE7:IFERR=17END
 1490 REPORT:PRINT" at line ";ERL
 1500 END
```

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

LOWER CASE FILE NAMES - Malo Hautus

The disc filing system does not distinguish between lower and upper case. If you have a program called 'fred' then saving another program called 'FRED' will overwrite the earlier one, and some versions of the DFS may leave the lower case 'fred' as the name stored in the catalogue. It can prove difficult to convert a lower case file name into the same upper case name. You have to *RENAME it to something different, and then *RENAME it back again.

BEEBCALC

OFFER STARTS ON 15th AUGUST

We are pleased to announce that we are now able to offer the new BEEBCALC spread sheet program for sale to members. Using BEEBCALC you can set up a table of figures or other information and specify the connections between these entries. Changing any one part of the display ensures that the rest of the display is immediately updated to match. It is particularly useful for exploring and analysing many financial applications including home and personal finance.

The program, which is in ROM, is produced by Computer Concepts (the suppliers of Wordwise). It is a powerful tool, with many features, including:

Full floating point maths; Use of both 40 and 80 column modes; Variable column width; Total cassette or disc system compatibility; Variable numbers of decimal places; Immediate re-calculation of values; Full printout without the need for special printer drivers.

In fact, it is very fast and simple to use. It is ideal for small business use, as well as being useful in the home and to the hobbyist.

[NOTE: BEEBCALC requires a series 1 O.S.]

The normal sale price for BEEBCALC is £40 (inclusive of VAT).
We are able to offer it to members at £35 fully inclusive of VAT and p&p.
Price to members outside UK is also £35 which includes the extra p&p but NOT VAT.
Cheques MUST be in 'Pounds Sterling'.

ADDRESS FOR BEEBCALC (Plus inclusive 1.2 ROM if ordered) Beebcalc Offer, PO BOX 50, St Albans, Herts, AL1 2AR.

THIS OFFER DOES NOT START UNTIL 15th AUGUST

WORDWISE Word Processor

******* N E W P R I C E N O W O N L Y £38 *******

This is a highly sophisticated word processing package for the BBC Micro, and compares favourably with those currently available on other microcomputers. It makes full use of the BBC Micro's advanced facilities, and text is typed and edited in the 40 column Teletext mode, saving memory, thus allowing it to be used with more or less any TV. Wordwise will work equally well on cassette or disc based systems, and it is easy to transfer files from cassette to disc if you upgrade at a later date.

The text for BEEBUG magazine is produced entirely using Wordwise. If you would like further details of the facilities and features offered by Wordwise please send an A5 SAE.

[NOTE: Wordwise requires a series 1 O.S.]

********* Wordwise now includes a free "TYPING TUTOR" program **********

The normal sale price of Wordwise is £46 (inclusive of VAT).
To BEEBUG members the price has been reduced even further than before to: £38 fully inclusive of VAT and p&p.
Price to members outside UK is also £38, this includes the extra p&p but NOT VAT.
Cheques MUST be in 'Pounds Sterling'.

ADDRESS FOR WORDWISE (Plus inclusive 1.2 ROM if ordered) Wordwise Offer, PO BOX 50, St Albans, Herts, AL1 2AR.

WORDWISE OR BEEBCALC PLUS 1.2 ROM

We can supply the 1.2 ROM with your Wordwise or Beebcalc order for an extra £5. (Total price for Wordwise + 1.2 ROM is £43. Beebcalc + 1.2 ROM is £40).
Once Wordwise and Beebcalc have been ordered we cannot later supply the 1.2 ROM at the special discount of £5.
Orders must be sent to the same address as for Wordwise/Beebcalc - PO Box 50, St Albans, Herts, AL1 AR.

MULTIPLE ORDERS: We can offer generous discounts on multiple orders of Wordwise or Beebcalc in excess of two, please send an SAE, together with your requirements, for a quotation.

IF YOU WRITE TO US

BACK ISSUES (Members only)

All back issues are kept in print (from April 1982). Send 90p per issue PLUS an A5 SAE to the subscriptions address. This offer is for members only, so it is ESSENTIAL to quote your membership number with your order. Please note that the BEEBUG Reference Card and BEEBUG supplements are not supplied with back issues.

NOTE OUR NEW SUBSCRIPTIONS ADDRESS

Subscriptions & Software Address

BEEBUG
PO BOX 109
Baker Street
High Wycombe
Bucks
HP11 2TD

SUBSCRIPTIONS

Send all applications for membership, subscription renewals, and subscription queries to the subscriptions address.

MEMBERSHIP COSTS:

£5.40 for 6 months (5 issues)
£9.90 for 1 year (10 issues)
European Membership £16 for 1 year.
Elsewhere (Postal zones)
Zone A £19, Zone B £21, Zone C £23

SOFTWARE AND ROM OFFER (Members only)

These are available from the subscription address, which is our NEW software address also. (Note that this does not apply to Wordwise or Beebcalc - in this instance please see magazine for details).

IDEAS, HINTS & TIPS, PROGRAMS, AND LONGER ARTICLES

Substantial articles are particularly welcome and we will pay around £25 per page for these, but in this case please give us warning of anything that you intend to write. In the case of material longer than a page, we would prefer this to be submitted on cassette or disc in machine readable form using "Wordwise", "Minitext Editor" or other means. If you use cassette, please include a backup copy at 300 baud.

Editorial Address

BEEBUG
PO Box 50
St Albans
Herts
AL1 2AR

We will also pay £10 for the best Hint or Tip that we publish, and £5 to the next best. Please send all editorial material to the editorial address opposite. If you require a reply it is essential to quote your membership number and enclose an SAE.

BEEBUG MAGAZINE is edited and produced by Sheridan Williams and Dr David Graham.
Technical Editor: Colin Opie. Production Editor: Phyllida Vanstone.
Technical Assistant: Alan Webster.
Thanks are due to John Yale, Adrian Calcraft, Tim Powys-Lybbe, Graham Greatrix and Matthew Rapier for assistance with this issue.

BEEBUG NEW ROM OFFER

A special arrangement has been agreed between Acorn and BEEBUG whereby BEEBUG members may obtain the Series One Machine Operating System in ROM at the price of £5.85 including VAT and post and packing.

The ROM will be supplied with fitting instructions to enable members to install it in their machine.

If the computer does not subsequently operate correctly, members may take their machines to an Acorn dealer for the upgrade to be tested, which will be done at a charge of £6.00 plus VAT. This charge will be waived if the ROM is found to have been defective. If the computer has been damaged during the installation process, the dealer will make a repair charge.

Please note that we cannot accept EPROM-based operating systems (0.1 or 1.0) in lieu of payment. This can only be performed by Acorn dealers or by Acorn's service centre at Feltham, and applies only to users of the 0.1 O.S.

ADDRESS FOR 1.2 O.S. IF ORDERED ON ITS OWN:- ROM Offer, BEEBUG, PO Box 109, High Wycombe, Bucks, HP11 2TD. PLEASE ALLOW 28 DAYS FOR DELIVERY.

MAGAZINE CASSETTE OFFER

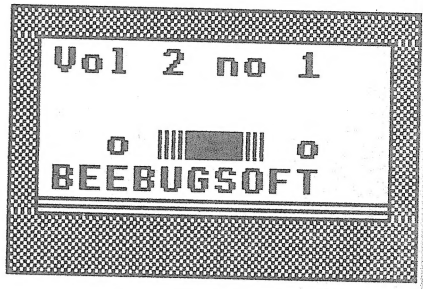
To save wear and tear on fingers and brain, we are offering each month a cassette of the programs featured in the latest edition of BEEBUG. The first program on each tape is a menu program, detailing the tape's contents, and allowing the selection of individual programs. The tapes are produced to a high technical standard by the process used for the BEEBUGSOFT range of titles. Ordering information, and details of currently available cassettes are given below.

Previous cassettes: Vol.1 No.10, Vol.2 No.1, Vol.2 No.2, Vol.2 No.3

This month's cassette (Vol.2 No.4) includes: Big Characters; *CODE and *LINE examples; Files Example; Model A and Model B Mars Lander; Spider's Web; Epson 8-Tone Dump (both article and program); Space Lords; Paintbox; Dual Screens; Two Programs Using the 6845; Birnbaum's Book Index; Cassette to Disc; and Crossword Answers.

For ordering information see BEEBUGSOFT advertisement at the back of this month's magazine supplement.

CASSETTE SUBSCRIPTION ADDRESS:
Please send a sterling cheque with order, together with your membership number and the date from which the subscription is to run, to:
Cassette Subscription, BEEBUG, PO Box 109 Baker Street, High Wycombe, Bucks, HP11 2TD.



MAGAZINE CASSETTE SUBSCRIPTION

We are now able to offer members subscriptions to our magazine cassettes. Subscriptions will be for a period of one year and are for ten consecutive issues of the cassette. If required, subscriptions may be backdated as far as Vol.1 No.10, which was the first issue available on cassette. This offer is available to members only, so when applying for subscription please write to the address below, quoting your membership number and the issue from which you would like your subscription to start.

CASSETTE SUBSCRIPTION PRICE:
UK £33 inc VAT and p&p
OVERSEAS (inc Eire) £39 inc p&p
(no VAT payable).

BEEBUG BINDER OFFER

BEEBUG MAGAZINE BINDER OFFER

A hard-backed binder for BEEBUG magazine is now available. These binders are dark blue in colour with 'BEEBUG' in gold lettering on the spine. They allow you to use the whole of the first volume of the magazine as a single reference book. Individual issues may be easily added or removed. The binders will also conveniently hold less than 10 issues, so that you can use it while you build up Volume 2 also.

BINDER PRICE
U.K. £3.90 inc p&p, and VAT.
Europe £4.90 inc p&p (VAT not charged)
Elsewhere £5.90 inc p&p
(VAT not charged)

Make cheques payable to BEEBUG.
Send to Binder Offer, BEEBUG, PO Box 109, Baker Street, High Wycombe, Bucks, HP11 2TD. Please allow 28 days for delivery on U.K. orders.

Please note that if you are ordering a binder now, you will not receive your binder until mid-August, we apologise for this, but have had to re-order.

Printed in England by Staples Printers St Albans Limited at The Priory Press.
ISSN 0263-7561